Modeling Appearances with Low-Rank SVM

Lior Wolf
School of Computer Science
Tel-Aviv University
wolf@cs.tau.ac.il

Hueihan Jhuang
CBCL
M.I.T
hueihan@mit.edu

Tamir Hazan
School of Engineering and CS
The Hebrew University of Jerusalem
tamir@cs.huji.ac.il

Abstract

Several authors have noticed that the common representation of images as vectors is sub-optimal. The process of vectorization eliminates spatial relations between some of the nearby image measurements and produces a vector of a dimension which is the product of the measurements' dimensions. It seems that images may be better represented when taking into account their structure as a 2D (or multi-D) array.

Our work bears similarities to recent work such as 2DPCA or Coupled Subspace Analysis in that we treat images as 2D arrays. The main difference, however, is that unlike previous work which separated representation from the discriminative learning stage, we achieve both by the same method.

Our framework, "Low-Rank separators", studies the use of a separating hyperplane which are constrained to have the structure of low-rank matrices. We first prove that the low-rank constraint provides preferable generalization properties. We then define two "Low-rank SVM problems" and propose algorithms to solve these. Finally, we provide supporting experimental evidence for the framework.

1. Introduction

The most natural representation of an image is a multi-dimensional array. The most frequently used learning algorithms, however, are designed for vector representation, that is, pixels of images are concatenated row after row into a long vector.

Viewed as a vector, an image no longer keeps its spatial relations between nearby pixels. Moreover, the dimension of the vector is the product of the array's dimensions, which can be extremely high, while the number of training examples remains small; problems associated with the curse of dimensionality such as overfitting are expected.

Feature selection techniques and dimensionality reduction methods (*e.g*. PCA) are used successfully to partly overcome the curse of dimensionality. For example, PCA and LDA are now common methods for face recognition applications. These methods, however, may perform better once domain information is incorporated.

Some recent research has incorporated the 2D structure of images into the dimensionality reduction process, such as 2DPCA [13], CSA [12] and 2DLDA [14]. Other work uses high order tensor decompositions to derive representations of image ensembles [5, 8, 9]. The ensemble's varying factors may include different faces, facial expressions, viewpoints, and illuminations. All above methods provide good performance, but learn the image representation separated from the supervised learning stage.

In this work we apply a unique variant of feature selection. Instead of selecting variables that best distinguish between the object classes, we select a set of rank-1 matrices. As we show below the variant method is tightly related to the original feature selection method. Let $X \in \mathbb{R}^{n\times m}$ be a data point as a matrix form, and $x \in \mathbb{R}^{nm}$ be its vectorized form. Let w and W be a separating hyperplane written as a vector and a matrix, then we use a linear decision rule of the form $\text{sign}(w^\top x + b) = \text{sign}(tr(WX^\top) + b)$.

Consider the feature selection scenario where a set of k variables with matrix coordinates $(\alpha_1, \beta_1)...(\alpha_k, \beta_k)$ is chosen. Each data-point X is then projected onto the linear space of the selected features. This subspace is spanned by matrices of the form $\mathbf{e}_{\beta_i}\hat{\mathbf{e}}_{\alpha_i}^\top$, where $\mathbf{e}_i$ is the i-th element of the standard basis of $\mathbb{R}^n$ and $\hat{\mathbf{e}}_j$ is the one of $\mathbb{R}^m$. Equivalently, we can constrain the matrix W to be of the form: $W = \sum_{i=1}^k d_i \mathbf{e}_{\beta_i}\hat{\mathbf{e}}_{\alpha_i}^\top$, for some $d_i \in \mathbb{R}$, and $i = 1, ..., k$.

The variant of feature selection we propose has a matrix W of a more general form: $W = \sum_{i=1}^k d_i \mathbf{u}_i \mathbf{v}_i^\top$, where $\mathbf{u}_i$ and $\mathbf{v}_i$ are arbitrary vectors, *i.e*. W is a low-rank matrix. This allows for basic features that form a spatial pattern rather than performing point-sampling. Note that the rank of W is at most k, and our analysis refers to the case where k is smaller than the matrix dimensions, otherwise, when $k \geq \min(n, m)$, W is not constrained. Also note that the decomposition of W into a sum of rank-one matrices is not unique. This ambiguity is reduced by requiring for the set of vectors $\{\mathbf{u}_i\}$ and $\{\mathbf{v}_i\}$ to be each orthogonal.

1-4244-1180-7/07/$25.00 ©2007 IEEE

In order to optimize for the low-rank matrix basis we define the Low-Rank SVM problem. Similar to SVM, our optimization criterion seeks a hyperplane which maximizes the margin from the decision boundary. In addition, we add the requirement that the separating hyperplane has to be a low-rank matrix. This results in a non-convex optimization problem, for which we offer iterative solutions. We provide two variants of the low-rank SVM problem. One is solved via cyclic optimization of one vector of parameters at a time. The other is an extension of a feature selection technique called AROM.

1.1. Related Work

Yang *et al*. [13] propose an algorithm called 2DPCA for face representation. As opposed to PCA, which treats images as vectors and finds their principal components, 2DPCA represents each image as a matrix, and maximizes the covariance of the projections of each row of measurements.

Xu *et al*. [12] propose an image reconstruction criterion to reconstruct the original image matrices using two low dimensional coupled subspaces, which encode the row and column subspaces of the image. They propose an iterative method, CSA (couples subspaces analysis) to optimize the criterion. They also prove that PCA and 2DPCA are special cases of the CSA.

Ye *et al*. [14] in the 2DLDA algorithm propose to use the 2D structure as a preprocess, to overcome the singularity problem of LDA which occurs when all the scatter matrices are singular.

The zero-norm linear-separation problem couples feature selection and separating hyperplane determination in one criterion. Let w be the separating hyperplane, the zero-norm of it is the number of non-zero elements in w. Given a set of training examples $\{x_i\}$ with corresponding labels $\{y_i\}$, the zero-norm SVM finds a hyperplane w and an offset b which optimize the following criterion:

$$\min \|w\|_0, \text{ subject to: } y_i(w \cdot x_i + b) \geq 1 \qquad (1)$$

Weston [11] shows that the problem can be solved by a simple procedure called AROM:

1. Train a linear SVM using the conventional ℓ_2 norm.
2. Weigh each data point by magnitude of the hyperplane w, *i.e.* $x_i \leftarrow D(w)x_i$, where $D(w)$ is the diagonal matrix of the elements of the hyperplane w.
3. Iterate the first 2 steps until convergence of w.

In each round, by re-weighting the data, important features gain more and more weighting, while the weights of less important features gradually decrease to zero. The algorithm works well for vector data, but not for matrix data, as we will demonstrate experimentally.

2. VC dim of a low-rank matrix separator

The overfitting caused by the curse of dimensionality can be quantified. Most naturally, one can estimate the expected generalization error, *i.e*. the difference between the testing error and training error. Bounds on the generalization error relate it to the capacity of the class of all possible classification rules. The VC-dimension [6] measures this capacity.

A classification model f_θ with some parameter vector θ is said to shatter a set of data points $\{x_1, x_2, \ldots, x_d\}$ if, for all possible assignments of labels to those points $\{y_1, y_2, \ldots, y_d\}$, there exists a θ such that $f_\theta(x_i) = y_i$. The VC dimension of a model f_θ is the maximum d such that there exists a set of data points of cardinality d which can be shattered by f_θ. For a p-dimensional linear model $f_{w,b}(x) = \text{sign}(w^\top x + b)$, the VC dimension equals $p + 1$. For linear models over vectorized $n \times m$ images, the VC-dimension is therefore $nm + 1$. We will show that low rank matrix separators have a much lower VC dimension.

We first investigate the hypothesis set $\mathcal{H}_1$ that corresponds to rank 1 matrices $\mathbf{u}\mathbf{v}^\top \in \mathbb{R}^{n\times m}$ and scalars $b \in \mathbb{R}$. Let $\{X_k\}$ be a set of data matrices with elements $X_k(i,j) \in \mathbb{R}$ and recall that the set $X_1, ..., X_d$ is shattered if there are 2^d pairs of hyperplanes and scalars in $\mathcal{H}_1$ which generate all the labels in $\{-1, 1\}^d$. The matrix $\mathbf{u}\mathbf{v}^\top$, the offset b and data X_k produce a label that equals $\text{sign}(f_k(\mathbf{u}, \mathbf{v}, b))$ whereas $f_k(\mathbf{u}, \mathbf{v}, b) = \sum_{i,j} X_k(i,j)u_i v_j + b$ is a degree 2 polynomial with $m + n + 1$ variables. The sequence $(\text{sign}(f_1(\mathbf{u}, \mathbf{v}, b)), \ldots, \text{sign}(f_d(\mathbf{u}, \mathbf{v}, b))) \in \{-1, 1\}^d$ is called a *sign pattern* of the polynomials $f_1, ..., f_d$. If the set $X_1, ..., X_d$ is shattered then all possible 2^d sign patterns are possible. We therefore have the following connection between sign patterns and VC-dimension:

Claim 1. *If for all d-tuple of polynomials $f_1, ..., f_d$ the number of sign patterns is less than 2^d then the VC-dimension is less than d.*

Claim 1 also applies for rank k hyperplanes. The important difference is that in this case the degree two polynomials $f_1, ..., f_d$ have $k(m + n) + 1$ variables. We state a general theorem due to Warren [10] that bounds the sign patterns of a set of polynomials. From this theorem and the above claim we derive a bound on the VC-dimension on $\mathcal{H}_k$ the set of all linear separators of rank at most k.

Theorem 1 (Warren '68)**.** *Let $f_1, ..., f_d$ be a sequence of d polynomials of degree at most h in q variables over the reals. Assume $d \geq q$ and $h \geq 1$, and let $e = exp(1)$, then the number of sign patterns is less than*

$$\left(\frac{4edh}{q}\right)^q \qquad (2)$$

Theorem 2 (VC-dim of $\mathcal{H}_k$)**.** *Let $\mathcal{H}_k$ be the hypothesis class of functions $g : \boldsymbol{R}^{n\times m} \rightarrow \{-1, 1\}$ of the form*

$g(X) = sign(tr(WX^\top) + b)$, where $W \in \boldsymbol{R}^{n\times m}$ is a rank k matrix and $b \in \boldsymbol{R}$. The VC-dim of the class $\mathcal{H}_k$ is less than $k(n+m)\log(k(n+m))$.

Proof: To show that the VC-dimension of $\mathcal{H}_k$ is less than $k(n+m)\log(k(n+m))$ we need to show that for every $k(m+n)+1$ variate polynomials $f_1, ..., f_d$ of degree at most 2 the log(number of sign patterns) is at most $d = k(m+n)\log(k(m+n))$, as:

$$\begin{aligned}\log\left(\frac{4ed\cdot 2}{k(m+n)}\right)^{k(m+n)+1} \leq \\ \leq (k(n+m)+1)\log\left(8e\cdot\log(k(n+m))\right) \\ \leq k(n+m)\log(k(n+m))\end{aligned}$$

whereas the last inequality holds for $k(m+n) \geq 100$. □

Theorem 2 proves that with respect to generalization error, the class $\mathcal{H}_k$ behaves, up to a logarithmic factor, similarly to hyperplanes of dimension $k(n+m)$. This is much lower than the $mn+1$ VC-dimension of separators which can be reshaped to general rank matrices.

3. Rank-k SVM

The VC-dimension result above holds for any linear separation obtained with a low rank matrix. If such a separator exists it may not be unique. On the other hand, an error free separator may not exist for a given data-set. We therefore suggest choosing a separator from the class of low-rank separators based on the SVM maximum margin principle [7]. The SVM optimization scheme is reduced to minimizing the sum of squares of the low rank separating hyperplane W, denoted be the Frobenius Norm $\|W\|_F^2 = \sum_{i,j} W_{i,j}^2$, with respect to linear constraints of the type $tr\left(W^\top X_i\right) - b \geq 1 - \xi_i$:

Problem 1 (Rank k SVM)**.** *Given a set of data points $X_1, X_2, \ldots, X_N$, and a set of corresponding labels $y_1, y_2, \ldots, y_N$, find W, b, ξ_i that optimize the following constrained minimization problem:*

$$\begin{aligned}\min_{W,b,\xi_i} \quad & \frac{1}{2}\|W\|_F^2 + C\sum_{k=1}^{l}\xi_i \\ \textit{subject to:} \quad & rank(W) \leq k \\ & y_i\left(tr(W^\top X_i) - b\right) \geq 1-\xi_i, \quad i = 1, ..., N \\ & \xi_i \geq 0\end{aligned}$$

The constraint $rank(W) \leq k$ is non-convex and the standard convex optimization techniques applied in vector SVM cannot be implemented. Below we describe an algorithm for solving problem 1.

3.1. Cyclic optimization algorithm

We constrain the rank of W to be k by representing W uniquely as the sum of k rank-1 orthogonal matrices $\mathbf{u}_r\mathbf{v}_r^\top$. Problem 1 can be equivalently formulated as:

$$\begin{aligned}\min_{\mathbf{u}_j,\mathbf{v}_j,b,\xi_i} \quad & \frac{1}{2}\sum_{j=1}^{k}\|\mathbf{u}_j\|_2^2\cdot\|\mathbf{v}_j\|_2^2 + C\sum_{k=1}^{l}\xi_i \\ \text{subject to} \quad & \mathbf{u}_{j_1}\perp\mathbf{u}_{j_2}, \quad \mathbf{v}_{j_1}\perp\mathbf{v}_{j_2} \quad 1\leq j_1<j_2\leq m \\ & y_i\left(\sum_{j=1}^{k}\mathbf{u}_j^\top X_i\mathbf{v}_j - b\right) \geq 1-\xi_i, \quad i=1,...,N \\ & \xi_i \geq 0\end{aligned}$$

Although the orthogonality constraints are non-convex they give rise to an alternating optimization scheme. We optimize in a cyclic manner over $\mathbf{u}_1, \mathbf{v}_1, \mathbf{u}_2, \mathbf{v}_2, ..., \mathbf{u}_k, \mathbf{v}_k$. When we optimize for a single vector of variables the rank k SVM is reduced to a vector-SVM and is solved by the standard SVM algorithm.

4. Low-Rank selection SVM

Similarly to the feature selection problem, where the optimal number of features may not be known in advance, it may be preferable to minimize directly the rank of the separating hyperplane. We therefore define the following optimization problem:

Problem 2 (Low-rank selection SVM)**.** *Given a set of data points $X_1, X_2, \ldots, X_N$, and a set of corresponding labels $y_1, y_2, \ldots, y_N$, find W, b, ξ_i that optimize the following constrained minimization problem:*

$$\begin{aligned}\min_{W,b,\xi_i} \quad & rank(W) + C\sum_{k=1}^{l}\xi_i \\ & \textit{subject to} \\ & y_i\left(tr(W^\top X_i) - b\right) \geq 1-\xi_i, \quad i=1,...,N \\ & \xi_i \geq 0\end{aligned}$$

4.1. Iterative weighting algorithm

In order to solve problem 2, we extend the AROM algorithm [11] to perform basis changes on top of feature selection. Let $U \in \mathbb{R}^{n\times n}$ and $V \in \mathbb{R}^{m\times m}$ be orthonormal matrices that represent some column and row bases. A matrix $X \in \mathbb{R}^{n\times m}$ can be represented in the U,V basis as

$$X^{uv} = U^\top XV. \tag{3}$$

i.e., instead of being represented in the usual basis $X = \sum_{i,j} X(i,j)e_i\hat{e}_j^\top$, where e_i and $\hat{e}_j$ are the standard bases of $\mathbb{R}^n$ and $\mathbb{R}^m$, it is represented as $X = \sum_{i,j} X^{uv}(i,j)\mathbf{u}_i\mathbf{v}_j^\top$, $\mathbf{u}_i$ and $\mathbf{v}_j$ being the columns of U and V.

Our iterative weighting algorithm is based on transforming the data-points by some new column and row bases U

and V, training an SVM in the new space and weighting the new features. The relationship between the separating hyperplane in the new basis W^{uv} and the one in the original basis is given by

$$W = UW^{uv}V^\top, \tag{4}$$

whereas for each X there holds,

$$\begin{aligned} tr(W^\top X) &= tr((UW^{uv}V^\top)^\top(UX^{uv}V^\top) \\ &= tr(VW^{uv}U^\top UX^{uv}V^\top) \\ &= tr(V^\top VW^{uv}X^{uv}) \\ &= tr(W^{uv}X^{uv}) \end{aligned}$$

Another way to observe Eq. 4 is to notice that the transformation $X \rightarrow U^\top XV$ is an orthogonal transformation[1] of X, and recalling that the dual transformation is the inverse transpose of the original one.

Below we describe the steps of the algorithm. Lower letters (*e.g.* x_i) describe the vectorized version of the corresponding upper letters (X_i).

1. Train SVM on x_i, y_i, $i = 1, ..., N$ and obtain w,b.
2. Reshape w as an $n \times m$ matrix W.
3. Use SVD to decompose W as UDV^T
4. $X_i^{uv} \leftarrow U^T X_i V$, $i = 1, \ldots, N$.
5. $D \leftarrow D + const$.
6. $X_i^{uv}(j,k) \leftarrow X_i^{uv}(j,k)D(j,k)$, for $i = 1, \ldots, N$, and $j = 1, \ldots, n$, and $k = 1, \ldots, m$.
7. $X_i \leftarrow UX_i^{uv}V^T$, $i = 1, \ldots, N$.
8. Go back to step 1 until convergence of w.

In step 1, we apply SVM to data and get a hyperplane w with nm dimensions. Then, in step 2, we reshape it as an $n \times m$ matrix, the same dimensions as the input matrices. In step 3, W is decomposed via SVD into two orthogonal matrices U and V, and one diagonal matrix D.

We now change basis to the space of rank-one matrices of the form $\{\mathbf{u}_i\mathbf{v}_j^\top | i = 1, \ldots, n, \text{ and } j = 1, \ldots, m\}$, where $\mathbf{u}_i$ is the i-th column of U, and similarly for $\mathbf{v}_j$ and V. This basis has the property, due to the properties of SVD, that for every k, the closest rank-k matrix approximation of W (in the least squares sense) is given by $\sum_{i=1}^{k} D(i,i)\mathbf{u}_i\mathbf{v}_i^\top$.

In this new basis $W^{uv} = U^\top WV = U^\top UDV^\top V = D$. Step 6, therefore, is equivalent to step 2 of AROM (see section 1.1). However, since D is a diagonal matrix, features of the form $\mathbf{u}_i\mathbf{v}_j^\top, i \neq j$ would be zeroed out even after one iteration if we use D as is. We therefore suggest to add a small constant to D in step 5 of the algorithm to allow the algorithm to converge more slowly to the desired feature space. After experimenting with synthetic data we fixed this value to be $D(1,1)/10$ in all of our experiments.

[1] because $tr\left((\mathbf{u}_i\mathbf{v}_j^\top)^\top \mathbf{u}_k\mathbf{v}_l^\top\right) = \delta_{ik}\delta_{jl}$

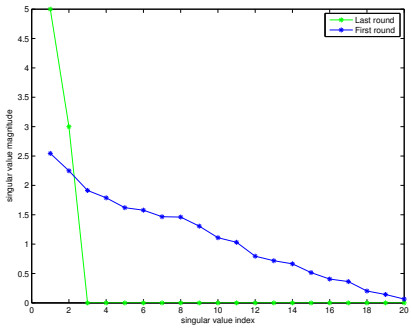

Figure 1. *The magnitude of diagonal terms of D in step 3 of the algorithm as they change through the iterations*

Finally, we transform the weighted feature matrices back to the original image space, and have the original data updated as

$$\begin{aligned} X_i &= UX_i^{uv}V^T \\ &= \sum_{k=1}^{n}\sum_{j=1}^{m} X_i^{uv}(j,k)\mathbf{u}_j\mathbf{v}_k^\top \\ &= \sum_{k=1}^{n}\sum_{j=1}^{m}[(d(j,k) + const)(\mathbf{u}_i^\top X_i\mathbf{v}_k)]\mathbf{u}_j\mathbf{v}_k^\top \end{aligned}$$

Fig. 1 shows the magnitude of diagonal terms of D in step 3, at the first iteration and the last iteration. The smooth curve shows at the first iteration, the magnitude of features slowly decrease and W has full rank. At the last iteration, only two relevant features are retained, and thus we get a W with rank 2.

4.2. Comparison with AROM

The feature selection algorithm AROM and our iterative weighting algorithm (sec. 4.1) both share a similar weighting scheme. The second one, however, is designed to work with matrices, and can select features that are rank-1 matrices instead of single variables. Also, while AROM is only invariant to a permutation of the variables, the iterative weighting low-rank SVM algorithm is invariant to orthogonal permutations of the row and column spaces.

Proposition 1. *Let A and B be arbitrary orthogonal matrices with dimensions $n_1 \times n$, and $m_1 \times m$ correspondingly, whereas $n_1 \geq n$ and $m_1 \geq m$. If a low-rank SVM algorithm results in a hyperplane W for the original training data X_i, $i = 1..N$, it will results in a hyperplane $\widetilde{W} = AWB^\top$ for the transformed training data $\widetilde{X}_i = AX_iB^\top$.*

Proof. The proposition follows from the fact (noted above in the explanation for Eq. 4) that the transformation $X \rightarrow AXB^\top$ is an orthogonal transformation of x (the vectorized version of X). The details are omitted for brevity. □

4.3. Extensions

Given a set of images containing multiple factors like color, viewpoint, and illumination, we can either put all the

images into a matrix, then apply the above algorithm, or represent them as a tensor in which each dimension corresponds to an image factor.

A natural extension of our algorithm to tensors is based on existing tensor decomposition methods [2, 1]. In step 3 of the algorithm we decompose W to a tensor product of a tensor S with orthogonal matrices $U_1, U_2, \ldots, U_M$, where the input examples are given as M-dimensional arrays. We will then transform each training example by U_j, for $j = 1 \ldots M$, and reweigh them according to the elements of S.

5. Relation to 2D decompositions

Assume that one uses an algorithm such as CSA [12] to learn the dominant row and the column spaces of the input samples X_1, X_2, projects the data to these spaces to get "noise-free" data and then learns. We show below that the resulting hyperplane would be of low-rank.

Proposition 2. *Assume that the columns and the rows of the data matrices $X_1, X_2, ...$ come from an m dimensional subspaces V_1, V_2 respectively, then the optimal hyperplane W^* is of rank m.*

Proof: Omitted for brevity.

6. Experiment

In this section we provide several experiments to support the low-rank learning framework. In our experiments the cyclic update algorithm in Sec. 3.1 converges much slowly than the iterative weighting algorithm in Sec. 4.1. We therefore did not run a full set of experiments for the cyclic update algorithm. We report the results for the iterative algorithm as follows.

Synthetic Data Using synthetic data, we compared the performance of SVM, AROM [11], 2DPCA [13] followed by AROM, and our proposed low-rank SVM iterative algorithm. The synthetic data was created in increasing levels of complexity. At first, we create vectors with dimension 100, and then arrange them into 10×10 matrices. Then, we made the structure of matrices more elaborate by altering the column and row space via orthogonal transformations. Lastly, we made the problem even more challenging by multiplying the data matrices from left and right by matrices of size 50×10 with orthogonal rows.

In the first synthetic data experiment, each data point is a vector in $\mathbb{R}^{100}$, in which the first six dimensions out of 100 are relevant. These vectors were drawn using the protocol given by Weston [11]: with a probability of 0.7, the first three features are drawn as $x(i) \sim y \cdot N(i, 1)$, and the second three features are drawn as $x(i) \sim N(0, 1)$. These points constitute the positive examples. The negative examples, which occurs with a probability of 0.3, have the first three variables drawn as $x(i) \sim N(0, 1)$ and the second three as $x(i) \sim y \cdot N(i - 3, 1)$. The remaining features are class independent noise drawn as $x(i) \sim N(0, 20)$, $i = 7, \ldots, 100$. In order to have similar distributions to relevant and to irrelevant variables, all variables are scaled to have mean zero and standard deviation one. The order of the variables is then permuted randomly. SVM and AROM used the vector form of the input data, while the other algorithms received a matrix form where the vectors were reshaped as 10×10 matrices X.

	original X	AXB^T A,B: 10×10	AXB^T A,B: 50×10
SVM	92.63 ± 1.69	92.63 ± 1.69	89.12 ± 3.72
AROM	99.90 ± 0.2	80.96 ± 8.01	75.54 ± 9.03
2DPCA+AROM	97.84 ± 1.11	97.84 ± 1.11	79.99 ± 8.07
LowRank	99.47 ± 0.35	99.47 ± 0.35	97.60 ± 2.48

Table 1. Synthetic experiments results (in percents). See Sec. 6.

The second and third synthetic data experiments were using data in which the relevant part consisted of a low-rank matrix subspace. Such matrices were generated by pre-multiplying the 10×10 matrices X (as above) by a random matrix with orthogonal columns A and post-multiplying them by a similar random matrix $B^\top$. A relevant feature $X(i, j)$ in the data is transformed into a general rank-1 matrix $X(i, j)a_i b_j^\top$, where a_i and b_j denote the $i - th$ and $j - th$ column of the matrices A and B.

Thus, with 6 relevant features in x, the ideal low-rank hyperplane for these experiments would be of rank-6. In the second experiment, the A and B were 10×10 orthogonal matrices. In the third experiment they were 50×10 matrices with orthogonal columns. These larger matrices, produce more input dimensions, thus potentially making the classification problem more difficult.

Table 1 summarizes the average recognition rate, and s.t.d (after the $\pm$ sign), for 20 repetitions, where for each repetition 500 points were generated. 50 points were used in the training stage and 450 points were used to evaluate the testing error. Our observations are (1) SVM has stable performance for all matrix transformations we used in this experiment, as expected. (2) AROM can help for the vector case, but not for the matrix cases. (3) 2DPCA+AROM works for some matrix transformations. (4) Low-rank SVM works well for both the vector and for the matrix cases.

Aligned face images We performed face detection experiments on $2,429$ face examples were taken from the CBCL face data set , and $2,500$ negative examples that were difficult non-faces extracted as false positives of a simple LDA-based face detector. Each example was represented as a 19×19 gray level images. We split the data-set 25% training, 75% testing and repeated the experiment 10 times. The average recognition rate for SVM was $92.7\% \pm 1.1\%$.

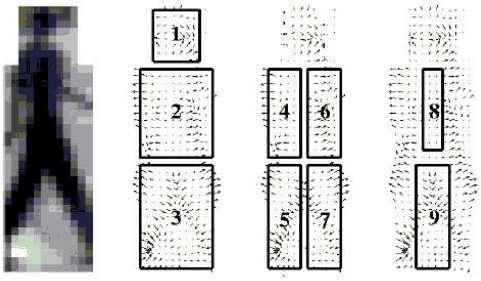

Figure 2. *The configuration of the nine sub-regions is displayed over the gradient image.*

AROM gave a comparable performance of $92.8\% \pm 1.0\%$. Low-rank SVM improves to $94.8\% \pm 1.2\%$.

Pedestrian detection We examined the performance of low-rank SVM on part-based representations dataset for pedestrian detection [4]. Each image was divided into 9 regions where local orientation statistics were generated with a total of 22 numbers per region (see Fig 2), thereby making a 22×9 matrix representation. There were 5,000 training and 10,000 testing examples, split evenly between positive and negative examples, and we repeated the experiment 10 times. The average recognition rate for SVM was 88.3%, whereas AROM achieved a comparable rate of 88.4%. Low-rank SVM improved this rate to 89.6%.

Action recognition We use the action data-set of [3], which contains six types of human actions, walking, jogging, running,boxing, hand waving and hand clapping, performed several times by 25 subjects in four different conditions: outdoors, outdoors with scale variation, outdoors with different clothes and indoors. The sequences were downsampled to the spatial resolution of 160x120 pixels and have a length of four seconds in average. As used in [16], sequences of 8 persons were used as training sets. The rest were used as the test set.

We computed five layers of features- a temporal gradient, I_t, two spatial gradient, I_x, I_y, and the two ratios between temporal and spatial gradients, $\frac{I_t}{I_x}, \frac{I_t}{I_y}$. These ratios implicitly contain velocity information arising from the constant brightness assumption.

Classification was performed at three frames at a time. The tensor representation of each example was therefore 3 frames $\times$ 5 layers $\times$ 160 $\times$ 120. We ran a series of binary tests, where in each case we classified one action as positive and another as negative. The average recognition rate for SVM was 75.8%, whereas AROM achieved a lower rate of 63.1%. Tensor Low-rank SVM improved the recognition rate significantly to 84.3%.

7. Conclusions

We presented the *low rank separators* framework that restrict the separating hyperplane to be a low rank matrix. We proved that the VC-dimension of this classification problem is significantly better than that of the general hyperplane classification problem, therefore the low-rank separating hyperplane can generalize better than a general hyperplane. We show that this is the case in synthetic statistical data and in several real-world data sets. Algorithmically, we find the low-rank separator by generalizing the zero-norm SVM optimization scheme to the matrix case, or more generally to the tensor case. In the future we would like to study the application of our methods to unordered variables and to other feature selection methods, such as boosting.

Acknowledgments

LW is supported by the Israel Science Foundation (grants No. 1440/06, 1214/06), and by the Colton Foundation.

References

[1] T.G. Kolda, Orthogonal tensor decompositions SIAM Journal on Matrix Analysis and Applications, 2001.

[2] L. De Lathauwer, B. De Moor, and J. Vandewalle, A multilinear singular value decomposition SIAM Journal of Matrix Analysis and Applications, 2000.

[3] C. Schuldt, I. Laptev and B. Caputo. Recognizing Human Actions: A Local SVM Approach, *ICPR*, 2004.

[4] A. Shashua et-al. Pedestrian Detection for Driving Assistance Systems. *IEEE Intelligent Vehicles Symposium*, 2004

[5] A. Shashua and A. Levin. Linear Image Coding for Regression and Classification using the Tensor-rank Principle. *CVPR*, 2001

[6] V. Vapnik and A. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. Theory of Probability and its Applications, 1971.

[7] V. Vapnik. The Nature of Statistical Learning Theory. Springer-Verlag, 1999.

[8] M.A.O. Vasilescu and D. Terzopoulos, Multilinear analysis of image ensembles: Tensorfaces ECCV, 2002

[9] H. Wang and N. Ahuja, Facial expression decomposition ICCV, 2003

[10] H.E. Warren, Lower bounds for approximation by non-linear manifolds. Trans. Amer. Math. Soc., 1968.

[11] J Weston, A Elissee, B Scholkopf and M Tipping, Use of the 0-norm with linear models and kernel methods JMLR, 2003.

[12] D. Xu, S. Yan, L. Zhang, Z.Liu, H.J. Zhang, Coupled subspaces analysis MSR-TR-2004-106,2004.

[13] J. Yang, D. Zhang, A. F. Frangi, and J. Yang, Two-dimensional PCA: a new approach to appearance-based face representation and recognition PAMI, 2004.

[14] J. Ye, R. Janardan, and Q. Li, Two-Dimensional Linear Discriminant Analysis NIPS, 2004.