Adaptive and Weighted Collaborative Representations for Image Classification

Radu Timofte[a,*], Luc Van Gool[a,b]

[a] *VISICS, ESAT-PSI / iMinds, University of Leuven, Belgium*
[b] *Computer Vision Lab, D-ITET, ETH Zurich, Switzerland*

Abstract

Recently, (Zhang et al., 2011) proposed a classifier based on collaborative representations (CR) with regularized least squares (CRC-RLS) for image face recognition. CRC-RLS can replace sparse representation (SR) based classification (SRC) as a simple and fast alternative. With SR resulting from an l_1-regularized least squares decomposition, CR starts from an l_2-regularized least squares formulation. Moreover, it has an algebraic solution.
We extend CRC-RLS to the case where the samples or features are weighted. Particularly, we consider weights based on the classification confidence for samples and the variance of feature channels. The weighted collaborative representation classifier (WCRC) improves the classification performance over that of the original formulation, while keeping the simplicity and the speed of the original CRC-RLS formulation. Moreover we investigate into query-adaptive WCRC formulations and kernelized extensions that show further performance improvements but come at the expense of increased computation time.

Keywords:
ridge regression, collaborative representation, sparse representation, classification, kernel

1. Introduction

For face recognition, (Wright et al., 2009) pointed out the effectiveness of representing newly observed faces as linear combinations of previously observed ones. Usually, the problem is formulated as one of finding the coefficients that minimize the residual between a new sample and its linear reconstruction. The residual is commonly measured as the least squares difference, which allows for an algebraic solution. When all previous faces or samples contribute to the optimal linear combination, one has a so-called Collaborative Representation (CR) (Zhang et al., 2011). Aside from the basic least squares criterion for the creation of such optima, other constraints have been considered as well. For stabilizing the coefficients of the least squares representation, one could add a term that tries to minimize the l_2-norm of the coefficients, while still admitting an algebraic solution. Enforcing sparsity on the solution leads to an l_0-regularization, i.e. an l_1-regularized least squares problem in practice, known as *lasso* (Tibshirani, 1996). Such adaptations yields a Sparse Representation (SR) (Wright et al., 2009), a key element in compressed sensing. Indeed, most signals admit a decomposition over a reduced set of signals from the same class (Bruckstein et al., 2009). Unfortunately, there no longer exists a known algebraic solution in that case. A combined l_1/l_2 regularization renders the coefficients more robust and enforces group sparsity (Zou and Hastie, 2005).

The resulting coefficients carry a meaning in that they reflect the importance of each sample. The coefficients and resulting residuals are used in the classification. The newly incoming sample or 'query' is assigned to the class that has the minimum residual error or the largest sum of coefficient magnitudes.

Here, we investigate the influence of weighting the samples and features in the aforementioned representations, in particular in the l_2-regularized least squares formulations with algebraic solutions. In real classification tasks the training samples are not equally discriminative. Moreover, the coefficients of some samples correlate more closely with the correct prediction by the

*Corresponding author. tel: +3216321718 (work) or +32494358850 (mobile)
Email addresses: radu.timofte@esat.kuleuven.be (Radu Timofte), vangool@vision.ee.ethz.ch (Luc Van Gool)
URL: http://homes.esat.kuleuven.be/~rtimofte/ (Radu Timofte)

classifier. The same holds for the feature channels, as some are more informative than others for the classification process. One may also let depend the weights on the particular query that is to be classified. We extend our previous work (Timofte and Van Gool, 2012b) by addressing such adaptive weightings and by investigating the corresponding kernel extensions, as a means for further gains in classification performance.

The remainder is organized as follows. Section 2 briefly reviews the least squares based formulations. Section 3 presents weighted variants, Section 4 adapts to query-specific weighting, and Section 5 discusses the kernel trick. In Section 6 we describe the classifiers based on these representations. Section 7 shows experimental results, while Section 8 concludes the paper.

Notations and assumptions. The training data is stacked column-wise forming a matrix $X = [x_1, x_2, \cdots, x_m] \in \mathbb{R}^{n \times m}$ with $m \in \mathbb{N}$ n-dimensional training samples. $\beta \in \mathbb{R}^m$ denotes the vector of coefficients from the linear representation of a query sample $y \in \mathbb{R}^n$ over the (training) samples (X). We assume each sample to be zero mean and to have unit length (the l_2-norm is 1). By $||x||_p$ we denote the l_p-norm of a vector x, $||x||$ is the Euclidean norm (or l_2-norm), X^T is the transpose of matrix X, and X^{-1} the inverse. $diag(x)$ is the diagonal matrix with the vector x on the diagonal. I is the identity matrix.

2. Least Squares formulations

We shortly review three of the best-known regularized least squares formulations.

The Ordinary Least Squares (OLS) solves:

$$\hat{\beta}_{OLS} = \arg\min_{\beta} ||y - X\beta||^2 \tag{1}$$

If $(X^T X)^{-1}$ exists we have as algebraic solution:

$$\hat{\beta}_{OLS} = (X^T X)^{-1} X^T y \tag{2}$$

The Collaborative Representation with Regularized Least Squares (CR) (Zhang et al., 2011), solves:

$$\hat{\beta}_{CR} = \arg\min_{\beta} ||y - X\beta||^2 + \lambda_{CR}||\beta||^2 \tag{3}$$

where $\lambda_{CR} \in \mathbb{R}$ is a regulatory parameter. The algebraic solution becomes:

$$\hat{\beta}_{CR} = (X^T X + \lambda_{CR} I)^{-1} X^T y \tag{4}$$

By adding the l_2 regularization we cope with the case where $X^T X$ is singular and, moreover, we stabilize/robustify the solution and make it less noise dependent.

The Sparse Representation (SR) (Wright et al., 2009) is obtained by enforcing sparsity by means of l_1-regularization instead of l_2 as for CR:

$$\hat{\beta}_{SR} = \arg\min_{\beta} ||y - X\beta||^2 + \lambda_{SR}||\beta||_1 \tag{5}$$

where $\lambda_{SR} \in \mathbb{R}$ is the Lagrangian regulatory parameter. For this problem, also known as *lasso* (Tibshirani, 1996), one does not have an algebraic solution, but efficient optimization solvers are available, like LARS (Efron et al., 2004), Feature Sign (Lee et al., 2006) or L1LS (Kim et al., 2007).

By combining sparsity and robustness by means of joint l_1 and l_2 regularization we pursue group sparsity:

$$\hat{\beta}_{EN} = \arg\min_{\beta} ||y - X\beta||^2 + \lambda_2||\beta||^2 + \lambda_1||\beta||_1 \tag{6}$$

a problem known as (naive) Elastic Net (EN) (Zou and Hastie, 2005), where $\lambda_{1,2}$ are regulatory scalar parameters. $(1 + \lambda_2)\hat{\beta}_{EN}$ gives the compensated EN solution.

3. Weighted Representations

In this section we review weighted extensions of the aforementioned least squares formulations.

Generalized Least Squares (GLS) generalizes the Ordinary Least Squares (OLS) for cases with unequal variances or correlations between the observations. If $y = X\beta_{GLS} + \epsilon$ with zero mean residuals, i.e. $E[\epsilon|X] = 0$, and their variance is $Var[\epsilon|X] = \Omega$, GLS minimizes their squared Mahalanobis length to estimate β_{GLS}:

$$\hat{\beta}_{GLS} = \arg\min_{\beta} (y - X\beta)^T \Omega^{-1} (y - X\beta) \tag{7}$$

$$\hat{\beta}_{GLS} = (X^T \Omega^{-1} X)^{-1} X^T \Omega^{-1} y \tag{8}$$

This simplifies to Weighted Least Squares (WLS) in case Ω is diagonal.

Ridge Regression (RR), also known as Tikhonov regularization (Tikhonov and Arsenin, 1977), solves:

$$\hat{\beta}_{RR} = \arg\min_{\beta} ||y - X\beta||^2 + ||\Gamma_{RR}\beta||^2$$

$$\hat{\beta}_{RR} = (X^T X + \Gamma_{RR}^T \Gamma_{RR})^{-1} X^T y \tag{9}$$

where to alleviate ill-posed problems, $\Gamma_{RR} \in \mathbb{R}^{m \times m}$ is suitably chosen. $\Gamma_{RR} \in \mathbb{R}^{m \times m}$ is the Tikhonov matrix and enables to weight samples differently. RR simplifies

to OLS or CR if Γ_{RR} is null or a scaled identity matrix, respectively.

A Generalized Weighted Collaborative Representation (WCR) can have the following formulation:

$$\hat{\beta}_{WCR} = \arg\min_{\beta}(y-X\beta)^T\Omega_{WCR}^{-1}(y-X\beta)+\|\Gamma_{WCR}\beta\|^2 \quad (10)$$

$$\hat{\beta}_{WCR} = (X^T\Omega_{WCR}^{-1}X + \Gamma_{WCR}^T\Gamma_{WCR})^{-1}X^T\Omega_{WCR}^{-1}y \quad (11)$$

Similar to GLS, Ω_{WCR} gives the importance of each dimension, and similar to RR, Γ_{WCR} modulates the importance of each sample in the solution.

Adding sparsity regularization (l_1-norm) to WCR leads to a Generalized Weighted Elastic Net (WEN) formulation:

$$\begin{aligned}\hat{\beta}_{WEN} \quad = \arg\min_{\beta} \quad & \{(y - X\beta)^T\Omega_{WEN}^{-1}(y - X\beta)+ \\ & \|\Gamma_{WEN}\beta\|^2 + \|\Lambda_{WEN}|\beta|\|_1\}\end{aligned} \quad (12)$$

where $\Lambda_{WEN} \in \mathbb{R}^{m\times m}$ expresses the importance of each sample for the l_1 term. By adding the sparsity regularization we lose the advantage of having a clean algebraic solution.

Other recent l_1-regularized methods weight the coefficients in relation to the covariances of the training samples, as in Weight Fused Lasso (Daye and Jeng, 2009) or Weight Fused Elastic Net (Fu and Xu, 2012). Other pairwise constraints are used in Group Lasso (Yuan and Lin, 2006), Pairwise Elastic Net (Lorbert et al., 2010), or Trace Lasso (Grave et al., 2011).

4. Adaptive Weighted Representations

In the previous section we formulated weighted representations. We can make a distinction between the way the weights are set in the representation formulation:

i. independent of the query

ii. adaptive to (dependent on) the query

Independence of the query is the default case for weighted representations. It means that the weights are learned (estimated) from the training data or set without considering the specificity of an arbitrary query. The dependent or adaptive approach uses the specificity of the input query in the computation of the weights, and thus adapts the weighted representation to the nature of the input.

When the weighting matrix (or residual variance in WLS) Ω is not directly known, it can be estimated, as in Feasible Generalized Least Squares (FGLS) (Little and Rubin, 2002), adapted to the query. Here we consider the case of GLS. First, we can use OLS and obtain the residuals u, and take for Ω the diagonal matrix of squared residuals, $diag(\hat{u}_{OLS})^2$, and estimate $\hat{\beta}_{FGLS_1}$:

$$\hat{u}_{OLS} = y - X\hat{\beta}_{OLS},\ \Omega_{OLS} = diag(\hat{u}_{OLS})^2 \quad (13)$$

$$\hat{\beta}_{FGLS_1} = (X^T\Omega_{OLS}^{-1}X)^{-1}X^T\Omega_{OLS}^{-1}y \quad (14)$$

leading to

$$\hat{u}_{FGLS_1} = y - X\hat{\beta}_{FGLS_1},\ \Omega_{FGLS_1} = diag(\hat{u}_{FGLS_1})^2$$

$$\hat{\beta}_{FGLS_2} = (X^T\Omega_{FGLS_1}^{-1}X)^{-1}X^T\Omega_{FGLS_1}^{-1}y \quad (15)$$

Under certain assumptions (Little and Rubin, 2002), Ω_{FGLS} will converge after a number of iterations. Note that the same iterative approach for estimating Ω can be applied to the other weighted least squares formulations (such as WCR or WEN).

Note that, in our experiments, we usually achieved better performance when we work directly with the residuals in absolute values ($\Omega_{OLS} = diag(|\hat{u}_{OLS}|)$, $\Omega_{FGLS_i} = diag(|\hat{u}_{FGLS_i}|)$), instead of taking the squared residuals as in the standard formulations used here.

Among recent adaptive weighted (sparse) representations, we review two: adaptive lasso and adaptive elastic net.

Adaptive lasso (Zou, 2006) or adaptive SR (aSR) solves:

$$\hat{\beta}_{aSR} = \arg\min_{\beta}\|y - X\beta\|^2 + \|\Lambda_{aSR}|\beta|\|_1 \quad (16)$$

where $\Lambda_{aSR} = \lambda_{aSR}diag(|\hat{\beta}_{CR}|^{-\nu})$, with $\nu \in \{0.5, 1, 2\}$ an extra parameter. With the change $X^* = X/(|\hat{\beta}_{CR}|^{-\nu})$ (dividing element-wise, $x_i^* = x_i/(|\hat{\beta}_{CR,i}|^{-\nu}), i = 1, 2, \cdots, m$), the solution is given by $\hat{\beta}_{aSR} = \hat{\beta}^*/(|\hat{\beta}_{CR}|^{-\nu})$ (element-wise) where $\hat{\beta}^*$ is the lasso (SR) solution:

$$\hat{\beta}^* = \arg\min_{\beta}\|y - X^*\beta\|^2 + \lambda_{aSR}\|\beta\|_1 \quad (17)$$

aSR imposes the coefficients from the RR decomposition of the query to obtain a grouping effect on the sparse solution of the original SR.

Adaptive Elastic Net (aEN) (Zou and Zhang, 2009) adds the l_2 regularization to the adaptive lasso, aSR:

$$\hat{\beta}_{aEN} = \arg\min_{\beta}\|y - X\beta\|^2 + \|\Lambda_{aEN}|\beta|\|_1 + \lambda_2\|\beta\|^2 \quad (18)$$

where $\Lambda_{aEN} \in \mathbb{R}^{m\times m}$ is taken similar to Λ_{aSR}.

5. Kernel extensions

5.1. Kernel trick and Hilbert space

For dealing with classification problems that are not linearly separable the kernel trick (Schölkopf and Smola, 2001) was introduced. The nonlinear problem can become linear in the high (even infinite) dimensional kernel feature space. The nonlinearity is handled by the kernel choice.

In the following we explore kernel extensions that are possible for most of the representations used in this paper. By using the kernel trick we expect to better adhere to the linearity assumption of the least squares models. For example, Kernel Sparse Representations have been proposed in (Zhang et al., 2012) and the Kernel Ridge Regression (Schölkopf and Smola, 2001) is a standard tool.

We assume a nonlinear mapping function, ϕ, from the original Euclidean space $\mathbb{R}^n$ to the Hilbert space $\mathcal{H}$, $\phi : \mathbb{R}^n \rightarrow \mathcal{H}$. $\mathcal{H}$ is often known as "reproducing kernel Hilbert space" (RKHS) corresponding to a Mercer kernel $k(\cdot,\cdot)$ (Schölkopf and Smola, 2001). Given $x, y \in \mathbb{R}^n$, we have $k(x,y) = \langle \phi(x), \phi(y) \rangle_{\mathcal{H}}$, where $\langle \cdot,\cdot \rangle_{\mathcal{H}}$ is the inner product in the kernel feature space $\mathcal{H}$. Among the best-known nonlinear kernel functions is the Gaussian kernel $k(x,y) = \exp(-\tau \|x-y\|^2)$, where τ is a scalar parameter.

Let $\Phi(X) = [\phi(x_1), \phi(x_2), \cdots, \phi(x_m)] \in \mathbb{R}^{l \times m}$ be, in the l-dimensional kernel space $\mathcal{H}$, the mapped data points $X = [x_1, x_2, \cdots, x_m] \in \mathbb{R}^{n \times m}$. Since l can be very high, we can employ a projection matrix to a lower dimensional embedding space. Another approach is to use the samples z as decompositions over the training samples by means of the kernel function: $k(\cdot, y) = [k(x_1, y), k(x_2, y), \cdots, k(x_m, y)]^T \in \mathbb{R}^{m \times 1}$. A projection matrix $R \in \mathbb{R}^{m \times d}$, $m \gg d$, can be used to further lower the dimensionality of the representation. R can be set by a random scheme or using other (learned) dimensionality reduction techniques as in (Zhang et al., 2012).

In the sequel, K represents the Gram matrix, $K_{i,j} = k(x_i, x_j)$, and R is not used in our experiments (is set as the identity matrix, I of size $m \times m$).

5.2. Kernel Representations

In this section we give straightforward kernelized formulations for RR, SR, and EN, thereby avoiding more lengthy derivations as sometimes found in the literature (Saunders et al., 1998; Suykens and Vandewalle, 1999; Zhang et al., 2012). The formulations do not use the label information, we are in an unsupervised mode.

Kernel Ridge Regression (KRR) is formulated and solved as follows:

$$\hat{\beta}_{KRR} = \arg\min_{\beta} \|\phi(y) - \Phi(X)\beta\|^2 + \lambda_{KRR}\|\beta\|^2 \quad (19)$$

$$\hat{\beta}_{KRR} = (K + \lambda_{KRR} I)^{-1} k(\cdot, y) \quad (20)$$

Note that this is an unsupervised KRR, and it has a slightly different formulation and interpretation than the ones commonly found in the machine learning literature (Saunders et al., 1998; Suykens and Vandewalle, 1999). y in our case is the new sample to linearly decompose and not a label target. The advantages of using a supervised regression are shown in (Suykens and Vandewalle, 1999). The most important ones concern the optimality conditions.

While in our experiments we use the KRR in the formulation introduced, in practice one could use a formulation on reduced representation dimensionality:

$$\hat{\beta}_{KRR} = \arg\min_{\beta} \|R^T k(\cdot, y) - R^T K\beta\|^2 + \lambda_{KRR}\|\beta\|^2 \quad (21)$$

Kernel Sparse Representation (KSR) (as in (Zhang et al., 2012)) solves:

$$\hat{\beta}_{KSR} = \arg\min_{\beta} \|R^T k(\cdot, y) - R^T K\beta\|^2 + \lambda_{KSR}\|\beta\|_1 \quad (22)$$

A Kernel Elastic Net (combining KRR and KSR) can be formulated as follows:

$$\hat{\beta}_{KEN} = \arg\min_{\beta} \|R^T k(\cdot, y) - R^T K\beta\|^2 + \lambda_1\|\beta\|_1 + \lambda_2\|\beta\|^2 \quad (23)$$

Of course, we do not need to work directly with the high-dimensional kernel space mappings but use the kernel function $k(\cdot,\cdot)$ instead, i.e. the explicit decomposition over training samples $k(\cdot, y)$. Moreover, when the number of training samples is large, we could also learn and apply a projection matrix (R) to reduce the dimensionality and the running time of the kernel extended regularized least squares formulations.

The Gram matrix (K) should be column-wise l_2-normalized to match the assumptions of least squares models and the solvers employed for KSR and KEN.

6. Classification

6.1. Sparse Representation - based Classifier

The information used for classification starting from the least squares formulations usually is the set of residuals for each class c (Wright et al., 2009):

$$r_c(y) = \|y - X_c\hat{\beta}_c\| \quad (24)$$

where $\hat{\beta}_c$ are the coefficients and X_c the samples for class c in the full representation of y as defined by the coefficients $\hat{\beta}$ and the training samples X. The classification decision is then taken as:

$$class(y) = \arg\min_c r_c(y). \quad (25)$$

The decision for the Sparse Representation-based Classifier (SRC) (Wright et al., 2009) is given by eq. (24) where $\hat{\beta} = \hat{\beta}_{SR}$. Another, faster, approach is to base the decision on the absolute values of the weights as in (Timofte and Van Gool, 2011):

$$w_c(y) = \|\hat{\beta}_c\|_1, \; class(y) = \arg\max_c w_c \quad (26)$$

6.2. Collaborative Representation - based Classifier

The Collaborative Representation-based Classifier with Regularized Least Squares (CRC), as introduced in (Zhang et al., 2011), uses eq. (4):

$$\hat{\beta}_{CR} = Py, \; P = (X^T X + \lambda_{CR} I)^{-1} X^T \quad (27)$$

and the regularized residuals are calculated as:

$$r_c(y) = \|y - X_c\hat{\beta}_c\| / \|\hat{\beta}_c\| \quad (28)$$

The decision of CRC is made as in eq. (25). The projection matrix, P, is independent of the query y and can be precomputed. This gives CRC a large computational advantage over SRC, which runs a query-dependent optimization.

The computation of P can be troublesome if the number of data samples (X) is much larger than the dimensionality of the data. It involves the computation of an inverse matrix of size equal to the number of training samples. One can then operate on the transposed data and compute the pseudoinverse (Penrose, 1955):

$$P = ((XX^T + \lambda_{CR} I)^{-1} X)^T \quad (29)$$

By adapting the computation of P using eq. (27) or eq. (29) we permit CRC to scale well with either a very high dimensionality of the data or very large datasets.

6.3. Weighted Collaborative Representation Classifier

For our WCR-based Classifier (WCRC) we inherit the regularized residuals approach of CRC where:

$$P = (X^T \Omega_{WCR}^{-1} X + \lambda_{WCR}(\kappa_1 I + \kappa_2 \Gamma_{WCR}^T \Gamma_{WCR}))^{-1} X^T \Omega_{WCR}^{-1} \quad (30)$$

One can use the procedure of FGLS to estimate Ω_{WCR}, by fixing Γ_{WCR}. If we have sufficient data, a good estimate for Ω_{WCR} is provided by the variance in the training data. However, in our experiments, using the standard deviation instead of the variance provides better performance. We use the standard deviation for estimating the diagonal matrix Ω_{WCR} for all the experiments.

We estimate Γ_{WCR} using the training data and cumulating the evidence per each sample for correct and incorrect classification participation. The weights correlate with the WCRC classifier decision.

Let t_i be the label class of the i-th training sample/column x_i of X. For each training sample x_i we cumulate the corresponding i-th squared coefficients $\hat{\beta}_i^2(x_j)$ of the representations for all training samples x_j. The representations are calculated using eq. (30) where $\Gamma_{WCR} = I$ and $\kappa_2 = 0$. For samples sharing the class t_i of x_i we cumulate these squared coefficients in θ_i^+, otherwise in θ_i^-:

$$\theta_i^+ = \sum_{j=1}^{m} [t_j = t_i] \hat{\beta}_i^2(x_j), \; \theta_i^- = \sum_{j=1}^{m} [t_j \neq t_i] \hat{\beta}_i^2(x_j) \quad (31)$$

Then, the weights are set using:

$$\Gamma_{WCR} = diag([\frac{\theta_1^+}{\theta_1^+ + \theta_1^-}, \cdots, \frac{\theta_m^+}{\theta_m^+ + \theta_m^-}]^{\frac{1}{2}}) \quad (32)$$

The working parameters are reduced to a single one, λ_{WCR}, by empirically setting κ_1 to the mean value of $X^T \Omega_{WCR}^{-1} X$, and κ_2 to 10 times this value, respectively.

Note that using the regularized residuals, eq. (28), instead of normal residuals, eq. (24), does not generally improve the W/CRC performance. When we work with discriminative data embeddings, usually the performance degrades. However, we use the regularized residuals for all our reported results.

6.4. Adaptive Weighted Collaborative Classifier

The weights can also be set adaptively, by using the locality assumption. That is, we assume, as (Waqas et al., 2013) does, that the farther the query is from a training sample the higher the penalty for using that sample in the representation ought to be. We call this classifier Adaptive WCRC (AWCRC) and we use the formulation of WCRC, with Ω_{AWCR} estimated as for WCR and Γ_{AWCR} adapted to the query sample y:

$$\Gamma_{AWCR}(y) = diag(\|x_1 - y\|, \|x_2 - y\|, \cdots, \|x_m - y\|) \quad (33)$$

For AWCR the projection matrix depends on y:

$$P(y) = (X^T \Omega^{-1} X + \lambda_{AWCR}(\kappa_1 I + \kappa_2 \Gamma(y)^T \Gamma(y)))^{-1} X^T \Omega^{-1} \quad (34)$$

where $\Omega = \Omega_{AWCR}$ and $\Gamma = \Gamma_{AWCR}$. Thus, AWCR loses the speed advantage of WCR where the projection matrix is independent of the query and can be precomputed. For AWCRC, κ_2 is set to the mean value of $X^T \Omega_{AWCR}^{-1} X$ and κ_1 to 0.0001 this value, leaving us with a single regulatory parameter λ_{AWCR}, still to set.

6.5. Kernelized Collaborative Classifiers

The kernelized extensions for both WCR and AWCR are quite straightforward. For simplicity, we consider the importance of the feature channels to be uniform, i.e. Ω is set to the identity I. The solution is given in the kernelized space by:

$$\hat{\beta}_K = (K + \lambda_K(\kappa_1 I + \kappa_2 \Gamma_K^T \Gamma_K))^{-1} k(\cdot, y) \qquad (35)$$

for a query y. The kernel we work with is the Gaussian kernel. The kernelized variants for CRC, WCRC, and AWCRC are called KCRC, KWCRC, and KAWCRC, respectively.

7. Experimental results

7.1. Benchmark setup

Datasets. We use the AR and PIE face datasets, the MNIST handwritten digit dataset and the GTSRB traffic sign dataset. For the AR dataset (Martinez and Benavente, 1998) we keep the same settings as in (Zhang et al., 2011; Wright et al., 2009). There are 100 individuals for a total of 700 training and 700 testing face images of 60×43 grayscale pixels. For the CMU PIE face dataset [1] we use the subset[2] from (Cai et al., 2007) with near frontal poses under different illuminations and expressions, up to 170 images per each of the 68 subjects. Furthermore, we randomly pick a partition with 700 training samples and 700 testing samples for our experiments. The MNIST handwritten digit dataset[3] contains 70,000 handwritten digit images with a split for training and testing (LeCun et al., 1998). As in (Waqas et al., 2013), we randomly select 50 training images for each of the 10 digits from the training set and 70 from the testing set. Complementary experiments are conducted on the GTSRB traffic signs dataset (Stallkamp et al., 2011). We have 43 classes and use all the 39209 training and 12630 testing images. With the exception of AR images, all the other images were cropped and resized to 28×28 grayscale pixels.

Features. We work directly on grayscale pixel values, on eigenfaces (as in (Zhang et al., 2011)), and on low-dimensional projections. The projections are obtained using regularized Linear Discriminant Analysis (LDA) (Fisher, 1938), regularized Sparse Representation based Linear Projections (SRLP) (Timofte and Van Gool, 2011), and regularized Iterative Nearest Neighbors (INN) Linear Projections (INNLP) (Timofte and Van Gool, 2012a). LDA maximizes the interclass variance while minimizing the intraclass variance. SRLP uses SR to let the projections capture important structural information in the data. INNLP uses the faster INN representation instead. The regularization parameter is set to 0.001 for all projection methods. For computing SR, in SRLP, we use the Feature Sign (Lee et al., 2006) solver with tolerance 0.04 and regularization parameter 0.01, if not mentioned otherwise. For INN, in INNLP, we set the regularization parameter to 0.05. All the features are l_2-normalized before and after projection.

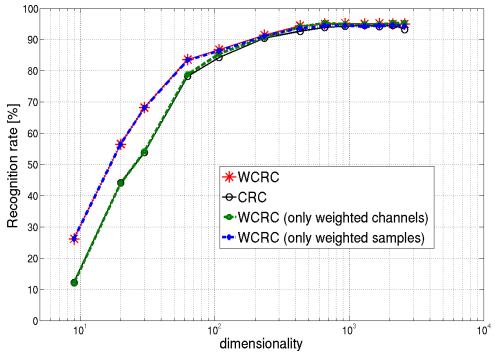


Figure 1: Dimensionality versus performance and weights on AR. The regulatory parameters are $\lambda = 0.001$ for WCRC and $\lambda = 0.01$ for CRC. The features are 9 up to 2580 grayscale pixel values from downscaled images.

Classifiers. Apart from the CR-based classifiers investigated here – CRC, WCRC, AWCRC, KCRC, KWCRC, and KAWCRC – we report results for the Nearest Neighbor (NN) classifier using the Euclidean distance, the Iterative Nearest Neighbors Classifier (INNC) (Timofte and Van Gool, 2012a) with $\lambda = 0.05$, the Sparse Representation-based Classifier (SRC) (Wright et al., 2009) with the Feature Sign (FeSg) solver (Lee et al., 2006) or the L1LS solver (Kim et al., 2007), and the standard Linear Support Vector Machines (LSVM) classifier. Most of the parameters were 4-fold cross-validated using 300 randomly picked training samples. For each parameter we considered up to 10 choices well spread over the range of values. For example, for the CRC and the SRC variants we considered $\lambda \in \{10^{-6}, 10^{-4} \cdots, 10^{-1}, 0.2, 0.5, 1, 5, 10\}$, and picked the one with the highest average classification rate for 10, 54, 99, and 300-dimensional features. In the experiments we fix the parameter for each classifier and dataset to the same value regardless the dimensionality

[1] http://www.ri.cmu.edu/projects/project_418.html

[2] http://www.zjucadcg.cn/dengcai/Data/FaceData.html

[3] http://yann.lecun.com/exdb/mnist/

of the features. We use the Gaussian kernel and empirically set $\tau = 0.2$ for the KCRC experiments and $\tau = 1$ for KSRC, respectively.

7.2. l_1, l_2 and data dimensionality

At the beginning, we investigate the role of l_1 and l_2 regularization in relation to the dimensionality of the data and the regulatory parameter. In Figure 2 we chose to depict the results in two important working regimes: low and high (relative to the features and projections) dimensional embeddings. Here we pick a low, 20-dimensional LDA embedding and a higher, 70-dimensional one. The classification rates are expressed as a function of the regulatory parameter for all the considered classifiers, *i.e.* λ_{WCR} is the parameter of WCRC. The Nearest Neighbor (NN) classifier does not have a corresponding regulatory parameter.

For high-dimensional data the l_2 regularization works very well, while the l_1 regularization is much more effective in the case of low-dimensional data. By weighting samples and/or channels/dimensions usually we improve the results over those of the initial formulations corresponding to uniform weights.

For assessing the importance of weighting channels or weighting samples individually or combined we use AR with grayscale value features and different dimensionality of the feature data. The results are depicted by Figure 1. The combined weighting is effective for low dimensions where sample weighting (when $\Omega_{WCR} = I$) dominates. For higher dimensions the channel weighting (when $k_2 = 0$) helps more, but the improvement is small. WCRC achieves 95.6%, $\sim$ 2% better than the best CRC results.

Our WCRC result on AR compares to the one recently reported by (Yang et al., 2012) for the Relaxed Collaborative Representation (RCR) Classifier (RCRC), 95.6%-WCRC vs. 95.9%-RCRC. RCRC approach groups the features (corresponding to blocks in the original images) and per each such group considers a different weighting scheme and its corresponding coding for an input query. The weighted solution for RCR is obtained through numeric iterations, while, in our case, WCR still has a direct algebraic solution. Under the same conditions, using Matlab scripts and tested on the AR dataset, with samples of 60×43 grayscale pixel values, our WCRC is more than 2.5 times faster than RCRC. We used the scripts provided by the authors.

7.3. Classification

The classification performance of the least squares based classifiers was tested on AR, MNIST, PIE, and GTSRB for different types of projections.

The AR results are in Table 1. WCRC consistently improves over CRC for low dimensional eigenfaces or projections (< 99), while for the higher dimensionalities the CRC and WCRC are on par. Regardless the features, for the lowest dimensionalities (< 30), the best classifiers are INNC and AWCRC, both relying strongly on the distance to the query information. In the medium range (≥ 30 and < 99) AWCRC and SRC are the best performers. For the higher dimensions (≥ 99) we do not have clear winners – CRC, WCRC, AWCRC, and SRC exhibit similar performance. Moreover, LSVM joins the group for discriminant projections. Using the kernel trick can further improve the performance especially for the low dimensionalities but at the expense of increased running times.

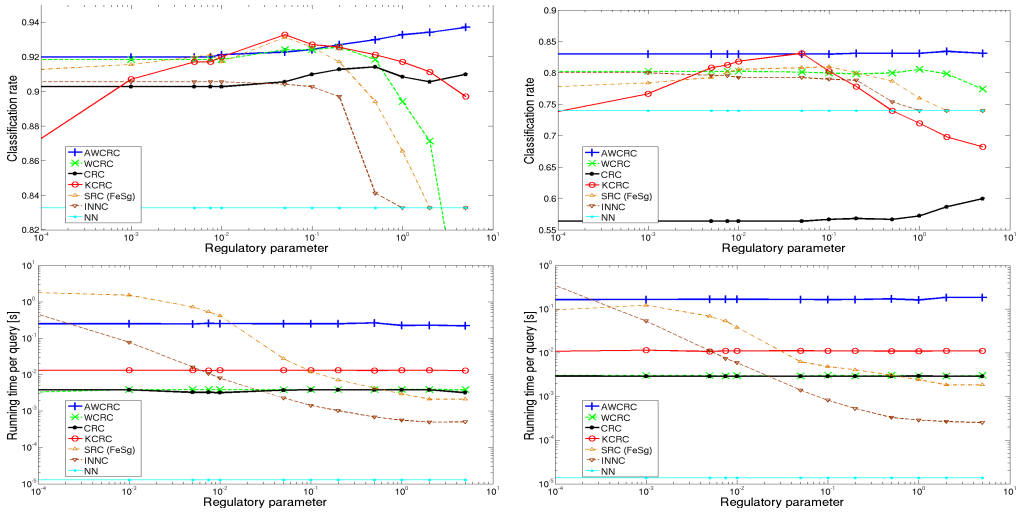
For GTSRB (see Table 2), we have a larger pool of training samples and WCRC comes out ahead of CRC. However, both methods are below the top SRC(FeSg) and INNC classifiers with the exception of when we use high-dimensional eigenfaces. GTSRB is 2 orders of magnitude larger than AR in terms of number of samples and combined with the relatively low-dimensional embeddings of the LDA, SRLP, and INNLP projections, seems to not accommodate W/CRC well.

If we contrast the results from AR and GTSRB, we see the importance of the sparsity (imposed by l_1-regularization in SRC or directly by weights in INNC). Enforcing sparsity helps in obtaining meaningful least squares decompositions at class level from large pools of data (GTSRB), while for smaller pools, l_2 seems sufficient.

Overall, the results on PIE match the ones obtained on AR (see Table 4). The relative performance of the classifiers is the same. This does not come as a surprise since both deal with faces with a balanced and similar number of classes/individuals (68 in PIE vs. 100 in AR).

For the MNIST handwritten digit dataset, we have a large gap between the results obtained with SRLP projected features and the ones using eigenfaces. Our results with eigenfaces are significantly better than the ones reported by (Waqas et al., 2013). We suspect that the main difference is due to the fact that we use the eigenfaces as in the Matlab code of (Zhang et al., 2011). We do not l_2-normalize the grayscale values prior to computing the eigenfaces and the mean face is not subtracted prior to the application of the eigenface projective matrix.

WCRC generally outperforms or is on par with the original CRC formulation in all our experiments. From the top classifiers, WCRC (and CRC) admits an algebraic solution and is faster than the SRC variants, but slower than NN and INNC, see Tables 1, 2 and 5.



a) 70 dim. LDA projection *b)* 20 dim. LDA projection

Figure 2: Classification rate and running time versus regulatory parameter on AR in the low and high dimensional regime using LDA projections.

Table 1: Face recognition rates [%] on AR.

	Classifier	λ	Dimensionality 5	10	30	54	99	120	300
eigenfaces	CRC	0.01	06.3	19.5	64.4	80.7	**89.6**	**90.4**	**93.9**
	WCRC	0.01	09.3	31.9	72.5	83.3	**89.3**	**90.6**	**93.6**
	$WCRC_s$	0.001	09.2	31.9	72.7	84.0	**89.1**	**90.1**	**93.9**
	$WCRC_c$	0.05	06.3	19.3	64.7	80.7	**89.3**	**90.3**	**93.9**
	AWCRC	0.01	**24.6**	**48.1**	73.7	82.4	87.1	88.6	92.3
	KCRC	0.0001	14.5	36.3	**78.6**	**85.0**	86.8	88.4	90.8
	KWCRC	0.0001	17.6	45.7	76.3	84.1	87.8	88.8	92.9
	KAWCRC	0.5	18.3	41.1	76.0	83.4	85.8	87.0	89.4
	SRC(L1LS)	0.01	06.2	19.6	64.7	81.0	**90.0**	**91.4**	**93.4**
	SRC(FeSg)	0.001	22.5	39.8	**73.7**	**84.1**	89.3	90.3	**93.6**
	KSRC(L1LS)	0.00001	11.4	29.0	68.0	79.1	87.7	89.4	90.8
	KSRC(FeSg)	0.00001	22.8	38.1	58.9	71.6	73.8	73.8	76.5
	INNC	0.01	**24.2**	**48.9**	69.0	74.1	77.3	77.6	79.6
	NN		23.5	43.4	59.1	68.1	69.8	70.4	71.4
	LSVM		10.3	33.5	68.2	76.3	81.3	82.4	84.5
LDA	CRC	1	10.3	26.8	75.1	90.4	**94.3**		
	WCRC	0.5	25.2	53.9	87.3	91.3	93.7		
	$WCRC_s$	0.1	25.2	53.5	86.3	92.4	94.0		
	$WCRC_c$	1	10.6	27.0	75.8	90.3	93.7		
	AWCRC	10	37.6	**62.4**	**89.0**	**92.9**	**94.6**		
	KCRC	0.05	35.2	56.5	88.4	92.4	93.0		
	KWCRC	0.001	34.2	55.7	**88.8**	**93.0**	**94.1**		
	KAWCRC	10	**38.3**	60.1	82.4	87.7	92.0		
	SRC(L1LS)	0.05	20.0	40.5	86.0	**92.8**	**94.8**		
	SRC(FeSg)	0.05	34.5	56.8	**87.1**	92.1	**94.4**		
	KSRC(L1LS)	0.001	32.1	54.2	**87.4**	91.7	93.4		
	KSRC(FeSg)	0.0001	34.8	59.1	84.8	91.0	**94.6**		
	INNC	0.01	**37.5**	**60.8**	86.6	88.8	92.6		
	NN		**37.2**	58.8	76.7	81.8	87.0		
	LSVM			47.2	80.7	90.4	93.6		
SRLP	CRC	1	10.2	28.6	77.0	90.0	93.7	**94.4**	93.9
	WCRC	0.2	24.8	55.4	86.3	91.1	**93.9**	93.9	94.0
	AWCRC	10	**38.6**	**63.0**	87.4	**92.6**	**94.4**	94.1	93.9
	KCRC	0.1	29.6	58.1	**88.3**	92.3	93.7	**94.7**	**94.4**
	KWCRC	0.05	34.5	57.9	**88.7**	**92.9**	**94.1**	**94.3**	**94.6**
	KAWCRC	10	37.2	61.5	82.6	88.0	91.9	**94.3**	90.4
	SRC(L1LS)	0.05	19.5	42.5	**86.8**	**92.6**	**94.6**	**94.7**	**94.7**
	SRC(FeSg)	0.05	34.9	60.0	86.0	**92.4**	**94.2**	**94.3**	**94.3**
	KSRC(L1LS)	0.001	31.0	57.5	**86.4**	91.7	93.7	93.8	93.6
	KSRC(FeSg)	0.0001	35.6	**61.7**	84.0	90.2	93.9	**94.4**	93.7
	INNC	0.01	**38.1**	**61.4**	85.6	89.1	91.5	93.1	92.7
	NN		37.1	59.9	76.4	82.6	86.0	86.3	87.0
	LSVM		33.0	50.1	83.0	90.7	93.6	94.1	93.8
INNLP	CRC	1	10.3	28.6	76.8	90.3	93.6	**94.3**	93.7
	WCRC	0.1	25.0	55.4	86.0	91.7	**93.7**	93.9	93.9
	AWCRC	2	**38.3**	**63.8**	**88.1**	**92.7**	**94.2**	**94.4**	**95.0**
	KCRC	0.1	29.2	57.8	88.0	92.4	93.6	**94.6**	**94.7**
	KWCRC	0.1	34.6	60.2	**88.6**	**92.9**	**93.7**	**94.6**	**94.6**
	KAWCRC	100	37.2	61.4	82.7	87.6	91.6	94.0	90.1
	SRC(L1LS)	0.05	18.9	42.2	**86.8**	**92.4**	**94.7**	**94.6**	**94.6**
	SRC(FeSg)	0.05	34.6	59.7	86.6	**92.4**	94.1	**94.1**	**94.4**
	KSRC(L1LS)	0.001	30.5	57.7	86.6	91.6	93.4	93.8	93.6
	KSRC(FeSg)	0.0001	34.9	**61.2**	83.8	90.6	94.1	**94.3**	94.0
	INNC	0.01	**38.5**	**61.5**	85.6	89.1	92.4	92.9	93.0
	NN		37.1	60.2	76.8	82.6	86.4	86.1	86.0
	LSVM		33.9	49.8	82.4	90.7	94.1	94.0	93.8

Table 2: Traffic sign classification rates [%] on GTSRB. *dims* stands for dimensions in the feature representation.

classifier	eigenf. (300dims)	SRLP (99dims)	INNLP (99dims)	LDA (42dims)
NN	66.05	89.54	89.35	91.84
WCRC	**86.12**	87.17	86.99	89.23
KCRC	84.06	84.39	84.41	83.20
CRC	84.34	83.43	83.56	84.08
INNC	77.70	**93.64**	**93.61**	**93.64**
SRC(FgSg)	85.31	**93.94**	**93.74**	92.91
SRC(L1LS)	74.78	79.34	79.41	93.01
LSVM	81.22	87.87	87.92	87.00
IKSVM	87.45	89.51	89.06	86.30
RBFSVM	81.34	92.43	92.57	92.46

Table 3: Handwritten digit recognition rates [%] on MNIST.

	Classifier	λ	Dimensionality 5	10	30	54	99	120	300
eigenfaces	CRC	5	52.4	73.7	89.6	89.6	89.9	90.0	89.7
	WCRC	5	57.1	77.3	89.1	90.0	89.9	89.7	89.0
	AWCRC	5	64.7	84.0	91.6	91.6	**91.6**	**91.6**	**91.0**
	KCRC	0.005	66.0	**87.1**	**93.7**	91.9	**91.7**	91.0	90.0
	KWCRC	0.005	**67.9**	85.3	**93.4**	**92.6**	**91.6**	**91.3**	90.4
	KAWCRC	5	65.3	**86.9**	**93.4**	**92.1**	**91.7**	**91.6**	90.4
	SRC(L1LS)	0.2	61.9	82.7	**93.0**	**91.9**	91.6	91.1	91.0
	SRC(FeSg)	0.001	62.3	82.3	91.1	91.4	90.9	91.3	91.1
	KSRC(L1LS)	0.001	**65.4**	83.7	91.6	91.7	91.4	91.6	91.1
	KSRC(FeSg)	0.001	62.4	83.7	91.4	**92.3**	**92.6**	**92.4**	**92.6**
	INNC	0.01	62.6	**84.4**	**92.9**	**92.0**	91.9	91.7	91.6
	NN		60.1	80.4	86.7	87.0	85.1	85.0	84.6
	LSVM		57.9	73.9	85.9	86.4	85.9	85.0	85.3
SRLP	CRC	100	43.9	57.7	51.1	54.1	56.9	55.7	59.4
	WCRC	100	46.0	56.0	56.9	59.1	62.7	**62.6**	61.5
	AWCRC	100	55.3	59.0	56.3	59.9	62.4	61.7	**63.3**
	KCRC	0.005	57.1	61.4	**63.3**	62.7	62.9	62.3	55.0
	KWCRC	5	**56.6**	**63.0**	62.3	62.1	62.1	**62.6**	62.3
	KAWCRC	1	**56.3**	62.4	**63.7**	**64.3**	**63.6**	**62.9**	55.6
	SRC(L1LS)	1	46.4	61.0	61.1	63.6	64.0	**65.3**	63.7
	SRC(FeSg)	0.5	49.6	61.0	60.6	62.1	63.6	**65.4**	61.3
	KSRC(L1LS)	0.5	**55.7**	**62.1**	**63.4**	**64.3**	**65.1**	**65.6**	64.3
	KSRC(FeSg)	0.5	**55.7**	61.3	**63.4**	63.1	64.4	65.0	**65.7**
	INNC	0.99	**56.1**	**62.0**	62.3	63.1	61.1	63.9	61.4
	NN		**56.1**	**62.0**	62.3	63.1	61.1	63.9	61.4
	LSVM		52.7	58.9	56.3	55.1	54.4	54.0	53.2

Table 4: Face recognition rates [%] on PIE.

	Classifier		Dimensionality						
		λ	5	10	30	54	99	120	300
eigenfaces	CRC	0.01	02.9	15.4	50.4	70.1	79.4	83.0	87.0
	WCRC	0.05	05.7	27.7	68.0	76.3	**83.6**	**84.3**	**87.6**
	AWCRC	0.0001	**09.1**	26.4	51.4	64.0	75.1	76.1	83.7
	KCRC	0.00001	07.6	36.0	**72.0**	**78.3**	**83.9**	**83.9**	83.3
	KWCRC	0.00001	08.6	**37.7**	71.3	**77.9**	82.0	83.0	86.4
	KAWCRC	0.01	**09.3**	28.1	64.4	74.4	79.9	80.1	81.6
	SRC(L1LS)	0.005	04.1	16.7	50.4	67.3	**80.6**	**83.6**	**87.1**
	SRC(FeSg)	0.005	07.3	22.1	50.0	67.9	80.1	82.7	86.4
	KSRC(L1LS)	0.00001	06.3	**27.6**	**62.6**	**71.4**	**80.7**	82.0	**86.9**
	KSRC(FeSg)	0.00001	07.0	20.0	53.1	64.3	70.4	70.9	73.6
	INNC	0.01	**09.1**	25.6	44.0	54.1	57.6	58.7	61.0
	NN		08.7	20.1	33.3	36.7	39.4	41.0	41.6
	LSVM		04.0	23.7	59.4	72.3	77.1	78.1	81.0
	Classifier		Dimensionality						
		λ	5	10	30	54	99	120	300
SRLP	CRC	100	24.6	54.7	82.9	88.1	**90.0**	**90.0**	89.3
	WCRC	2	41.6	72.3	85.1	86.1	86.7	87.1	88.7
	AWCRC	100	**58.4**	**78.1**	**88.7**	**90.3**	**90.1**	**90.3**	**90.7**
	KCRC	0.005	46.7	72.9	87.4	89.4	89.6	88.9	87.3
	KWCRC	0.5	49.0	68.1	87.1	88.7	89.4	**89.8**	**90.2**
	KAWCRC	10	56.0	76.9	**89.0**	89.4	**90.3**	**89.9**	88.0
	SRC(L1LS)	0.2	40.3	70.4	87.1	89.1	90.1	90.3	90.7
	SRC(FeSg)	0.001	55.7	74.1	85.9	87.9	89.1	89.6	90.4
	KSRC(L1LS)	0.01	40.7	67.9	86.9	88.9	89.4	88.7	89.1
	KSRC(FeSg)	0.2	54.7	75.6	87.7	89.6	**91.3**	**91.3**	**92.0**
	INNC	0.01	**55.7**	**78.9**	**88.3**	**90.3**	**90.9**	90.6	90.4
	NN		55.1	**78.4**	87.3	88.3	89.1	88.7	88.4
	LSVM		50.6	67.9	85.4	88.0	88.4	87.7	88.4

7.4. Complexity and Running time

Speed can be a critical factor in practice. We report some of the recognition rates and the average running times per query in Table 5 for AR with 300-dimensional eigenfaces and GTSRB with 42-dimensional LDA embeddings. We employ Matlab scripts and we use the same system with Intel Core i7 CPU 860 @ 2.80GHz, 8 GBytes RAM, and Fedora for our experiments. For each setting, we run 10 times on the randomized test dataset and take the average running time corresponding to a single query sample. The influence on running time of the data dimensionality and regulatory parameter is revealed also in Figure 2. WCRC and CRC are very fast, orders of magnitude faster than the SRC formulation. The improvement obtained using the AWCRC formulation and/or the kernel trick more than doubles the running time of the classifier. Moreover, INNC (Timofte and Van Gool, 2012a) is faster than both CRC and WCRC but performs poorer for high-dimensional data.

Our empirical running time experiments match the time complexity of the methods. We consider m samples with n-dimensional representations. The time complexity for W/CRC in test is $O(mn)$, while the time complexity of AWCRC is dominated by solving eq. (34), which involves computation of an inverse for an $m \times m$ matrix, thus, at least $O(m^{2.373})$. The time complexity for INNC is $O(kmn)$, where k is the number of iterations. The hidden constants and terms in the time complexity of W/CRC are bigger than those of INNC. In practice, for k reasonably small, INNC proves to be the faster method (see Table 5, INNC has $\lambda = 0.05, k = 62$ for AR and $\lambda = 0.3, k = 12$ for GTSRB). In comparison, SRC requires optimization. The Feature Sign solver (Lee et al., 2006) is efficient, but its computational burden is higher than the one of the W/CRC or INNC methods.

Table 5: Average running times per query on AR and GTSRB.

	AR w/ Eigenfaces(300)		GTSRB w/ LDA(42)	
classifier	Recog.[%]	Time (s)	Recog.[%]	Time (s)
NN	71.43	0.0001	91.84	0.0009
INNC	79.54	0.0023	**93.64**	**0.0081**
SRC(L1LS)	93.41	0.8187	93.01	2.8524
SRC(FeSg)	93.56	0.2175	92.91	0.1679
CRC	93.76	0.0044	84.08	0.0381
WCRC	**93.72**	**0.0044**	89.23	0.0392
KCRC	88.27	0.0121	88.52	0.1628
AWCRC	93.12	0.2150	–	–

7.5. Adaptive, weighted, and/or kernelized?

The weighted variant (WCRC) can improve the results over those of the original CRC formulation, as expected since it includes either discriminant or correlation information extracted from the training samples. The weighting better fits the classification task.

The adaptive weighting (AWCRC) is found to help accuracy-wise especially for the low-dimensional spaces where the extra adaptive information (distance to the sample in our case) helps to obtain more discriminative representations. If NN is better than CRC or WCRC, then it is likely that using the distance to the query in the AWCRC formulation will boost the performance to at least the level of NN. However, in our experiments we found little or no improvement whenever we used the adaptive scheme estimation for the channel weights (the Ω weights). The biggest drawback of the query adaptive approaches is the computational overhead. They do not allow for the prior computation of a projective matrix as in the query-independent case.

As already proved for SVMs, using kernels usually helps the classification task. The improvement is visible especially for low-dimensional original spaces and for non-adaptive kernelized variants (KCRC and KWCRC). Using the kernel trick comes at a price – the computation time is more than double that of the original formulations, in our settings.

While in our experiments we fixed a couple of weighting strategies and settings for the sake of a clean exposition, one can easily imagine other such settings tailored to particular applications. Combining different weighting strategies is also possible, as well as the fine-tuning of the parameters.

In future work, we plan to devise methods to learn the (optimal) weights for the classification task at hand.

8. Conclusions

In this paper we reviewed current least squares-based representations and investigated the impact of adding weights and kernelization. We focused on the Weighted Collaborative Representation (WCR), spelling out out strong points as well weaknesses for image classification as part of WCR-based Classifiers (WCRC). WCR shares the simplicity and the effectiveness of the original CR formulation. The weights for WCR are computed offline and it still has an algebraic solution.

For validation we used face, digit, and traffic sign datasets. Out of these experimental results and keeping in mind a good speed-performance tradeoff the following picture emerges: for the lowest dimensions one can best use INNC, for somewhat higher dimensions SRC (Feature Sign) is the best performer, then to be replaced for high dimensions by W/CRC as the method of choice.

The adaptive weighting to the query (AWCRC) generally improves over the weighting independent of the query (WCRC), but heavily affects the running time.

Using the kernel trick significant improvements can be achieved over the flat formulations, especially for low dimensional embeddings and, again, at the expense of increased computation.

Acknowledgments. This work was partly supported by the European Commission FP7 ICT-269980 AXES project and the IWT/SBO ALAMIRE project.

References

Bruckstein A, Donoho D, Elad M. From sparse solutions of systems of equations to sparse modeling of signals and images. SIAM Review 2009;51(1):34–81.

Cai D, He X, Han J. Efficient kernel discriminant analysis via spectral regression. In: International Conference on Data Mining (ICDM'07). 2007. .

Daye ZJ, Jeng XJ. Shrinkage and model selection with correlated variables via weighted fusion. Computational Statistics and Data Analysis 2009;53(4):1284 –98.

Efron B, Hastie T, Johnstone I, Tibshirani R. Least angle regression. The Annals of Statistics 2004;32(2):407–99.

Fisher R. The statistical utilization of multiple measurements. Annals of Eugenics 1938;8:376–86.

Fu GH, Xu QS. Grouping variable selection by weight fused elastic net for multi-collinear data. Communications in Statistics - Simulation and Computation 2012;41(2):205–21.

Grave E, Obozinski G, Bach F. Trace lasso: a trace norm regularization for correlated designs. In: Advances in Neural Information Processing Systems (NIPS). 2011. p. 2187–95.

Kim S, Koh K, Lustig M, Boyd S, Gorinevsky D. An interior-point method for large-scale l_1-regularized least squares. Selected Topics in Signal Processing, IEEE Journal of 2007;1(4):606–17.

LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. Proceedings of the IEEE 1998;86(11):2278–324.

Lee H, Battle A, Raina R, Ng AY. Efficient sparse coding algorithms. In: Schölkopf B, Platt JC, Hoffman T, editors. NIPS. MIT Press; 2006. p. 801–8.

Little RJA, Rubin DB. Statistical Analysis with Missing Data (2nd ed.). John Wiley & Sons, Inc., 2002.

Lorbert A, Eis D, Kostina V, Blei DM, Ramadge PJ. Exploiting covariate similarity in sparse regression via the pairwise elastic net. JMLR - Proceedings of the 13th International Conference on Artificial Intelligence and Statistics 2010;9:477–84.

Martinez A, Benavente R. The AR face database. Technical Report; CVC Tech. Report No. 24; 1998.

Penrose R. A generalized inverse for matrices. Mathematical Proceedings of the Cambridge Philosophical Society 1955;51:406–13.

Saunders C, Gammerman A, Vovk V. Ridge regression learning algorithm in dual variables. In: Proc. of the 15th Int. Conf. on Machine Learning (ICML-98). Madison-Wisconsin; 1998. p. 515–21.

Schölkopf B, Smola AJ. Learning with kernels: support vector machines, regularization, optimization, and beyond. The MIT Press, 2001.

Stallkamp J, Schlipsing M, Salmen J, Igel C. The German Traffic Sign Recognition Benchmark: A multi-class classification competition. In: International Joint Conference on Neural Networks (IJCNN). 2011. .

Suykens J, Vandewalle J. Least squares support vector machine classifiers. Neural Processing Letters 1999;9(3):293–300.

Tibshirani R. Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society, Series B 1996;58(1):267–88.

Tikhonov AN, Arsenin VY. Solution of Ill-posed Problems. Winston & Sons, 1977.

Timofte R, Van Gool LJ. Sparse representation based projections. In: British Machine Vision Conference (BMVC). 2011. .

Timofte R, Van Gool LJ. Iterative nearest neighbors for classification and dimensionality reduction. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2012a. p. 2456–63.

Timofte R, Van Gool LJ. Weighted collaborative representation and classification of images. In: International Conference on Pattern Recognition (ICPR). IEEE; 2012b. p. 1606–10.

Waqas J, Yi Z, Zhang L. Collaborative neighbor representation based classification using l_2-minimization approach. Pattern Recognition Letters 2013;34:201–8.

Wright J, Yang AY, Ganesh A, Sastry SS, Ma Y. Robust face recognition via sparse representation. IEEE Trans Pattern Anal Mach Intell 2009;31(2):210–27.

Yang M, Zhang L, Zhang D, Wang S. Relaxed collaborative representation for pattern classification. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2012. p. 2224–31.

Yuan M, Lin Y. Model selection and estimation in regression with grouped variables. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 2006;68(1):49–67.

Zhang L, Yang M, Feng X. Sparse representation or collaborative representation: Which helps face recognition? In: ICCV. 2011. .

Zhang L, Zhou WD, Chang PC, Liu J, Yan Z, Wang T, Li FZ. Kernel sparse representation-based classifier. Signal Processing, IEEE Transactions on 2012;60(4):1684–95.

Zou H. The Adaptive Lasso and Its Oracle Properties. Journal of the American Statistical Association 2006;101(476):1418–29.

Zou H, Hastie T. Regularization and variable selection via the elastic net. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 2005;67(2):301–20.

Zou H, Zhang HH. On the adaptive elastic-net with a diverging number of parameters. Annals of statistics 2009;37(4):1733–51.