Learned Collaborative Representations for image classification

Jiqing Wu Radu Timofte Luc Van Gool
Computer Vision Lab, D-ITET, ETH Zürich, Switzerland

Abstract

The collaborative representation-based classifier (CRC) is proposed as an alternative to the sparse representation-based classfier (SRC) for image face recognition. CRC solves an l_2-regularized least squares formulation, with algebraic solution, while SRC optimizes over an l_1-regularized least squares problem. As an extension of CRC, the weighted collaborative representation-based classifier (WCRC) is further proposed. The weights in WCRC are picked intuitively, it remains unclear why such choice of weights works and how we optimize those weights. In this paper, we propose a learned collaborative representation-based classifier (LCRC) and attempt to answer the above questions. Our learning technique is based on the fixed point theorem and we use a weights formulation similar to WCRC as the starting point. Through extensive experiments on face datasets we show that the learning procedure is stable and convergent, and that LCRC is able to improve in performance over CRC and WCRC, while keeping the same computational efficiency at test.

1. Introduction

In the realm of face recognition, Ordinary Least Squares (OLS) formulation is a widely used strategy to solve the recognition problem. By minimizing the residues between a testing face and its linear combination of training faces we conclude which class it belongs to. Of course, in many situations due to its limitation researchers are motivated to propose more sophisticated methods by adding an l_n-regularization term to increase the recognition rate. To enforce the sparsity on solutions Wright *et al*. [14] apply l_1 norm as the regularization part of OLS formulation, which unfortunately does not have an algebraic solution. They obtain a sparse representation (SR) and a sparse representation-based classfier (SRC). As an alternative to SRC, Zhang *et al*. [15] proposes the l_2-regularization, obtaining a collaborative representation (CR) and a collaborative representation-based classifier (CRC). By CRC we can avoid the singular matrix appeared in the OLS method, stabilize the solution and decrease the importance of noisy samples. Besides, unlike SRC we still have an algebraic solution. By combining l_1- and l_2- regularized terms Zou *et al*. [18] proposes an Elastic Net (EN) problem to make use of both advantages of CR and SR. The solution (coefficients) of above methods reveals the importance of each training sample. For better reflecting the relation between a testing face and training faces, Timofte and Van Gool [11, 12] proposes the WCRC method, in particularly investigates l_2-regularized least squares with additional Tikhonov regularization (Tikhonov *et al*. [8]). Moreover, the effects caused by weighting samples and features are argued in details and a collection of selected weights is proposed.

On the other hand, recognition tasks always confront non-linear situation. Under such circumstances we often introduce a non-linear map from euclidean space to a Hilbert space and employ the so-called kernel trick (Schölkopf *et al*. [6]). Therefore, it is natural to extend mentioned methods such as Sparse Representation (SR), Elastic Net (EN) and Ridge Regression (RR) to kernel-based formulations KSR (Zhang *et al*. [16]), KRR (Saunders *et al*. [5]; Suykens *et al*. [7]), and KEN (Timofte and Van Gool [12]).

In this paper, we extend WCRC to a robust, solid mathematically founded method – learned collaborative representation-based classifier (LCRC). Instead of intuitively determining the weights, we use fixed point theorem [1] to optimize them by starting from the setup similar to WCRC. In addition, we attempt to fully exploit the information of training samples, without employing query-adapted technique. During our discussion we will explain why our WCRC-extended method LCRC works well and provide the evidence showing improvements over both CRC and WCRC.

Our paper is organized as follows. Section 2 shortly reviews CRC and WCRC formulations, then introduces LCRC, our proposed method. Section 3 refines the implementation details for learning the weights in LCRC and provides a pseudo-code of the learning algorithm. Section 4 presents the experimental setup and discusses the parameters and results over different face datasets and features. Section 5 summarizes the paper.

2. Collaborative Representation Classifiers

In this section, we briefly review collaborative representation-based classifier (CRC) and weighted collaborative representation-based classifier (WCRC). Then, we introduce our new method – learned collaborative representation-based classifier (LCRC).

Unless stated otherwise, we use the same notations and assumptions as Timofte and Van Gool [12]. Let $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M) \in \mathbb{R}^{N \times M}$ be the collection of M column-wise N-dimensional training samples, $\mathbf{x}_j \in \mathbb{R}^N$. The samples belong to K classes with labels in $\mathbf{C} = \{c_1, c_2, \ldots, c_K\}$. Let $\boldsymbol{\beta} \in \mathbb{R}^M$ be the coefficients of a linear combination of training samples regarding a query $\mathbf{y} \in \mathbb{R}^N$. $\|.\|$ marks the Euclidean norm, $\mathbf{I}$ the identity matrix, $\mathbf{X}^T$ the transpose of $\mathbf{X}$, and $\text{diag}(\mathbf{x})$ is the diagonal matrix with the vector $\mathbf{x}$ on the diagonal. We also assume that each training sample has zero mean and unit length.

2.1. CRC

Zhang *et al*. [15] proposed the Collaborative Representation-based Classifier with Regularized Least Squares (CRC) as an alternative to SRC. CRC first solves:

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\text{argmin}}\{\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda\|\boldsymbol{\beta}\|^2\} \tag{1}$$

with a regulatory parameter λ. The algebraic solution is:

$$\hat{\boldsymbol{\beta}} = \mathbf{P}\mathbf{y}, \mathbf{P} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T \tag{2}$$

where $\mathbf{P}$ is precomputed offline for a given $\mathbf{X}$ and λ.

The classification decision of CRC is given by:

$$\text{class}(\mathbf{y}) = \underset{c \in \{c_1, c_2, \ldots, c_K\}}{\text{argmin}} \{\|\mathbf{y} - \mathbf{X}_c\hat{\boldsymbol{\beta}}_c\| / \|\hat{\boldsymbol{\beta}}_c\|\} \tag{3}$$

where $\mathbf{X}_c$ is the collection of training samples belonging to class c and $\hat{\boldsymbol{\beta}}_c$ are the corresponding coefficients.

2.2. WCRC

Often there is a difference in how helpful is each training sample in classification. This is due to how specific is the sample to its own class and how discriminative is from other class samples. Moreover, the samples are described by features (*e.g*. pixel values) and these features again can discriminate differently among each other, for instance, the image pixels/regions on the face are more useful in classification than those on the background. Such arguments lead to the Weighted Collaborative Representation-based Classifier (WCRC) proposed by Timofte and Van Gool [12].

WCRC solves:

$$\begin{aligned}\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\text{argmin}}\{&(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^T\boldsymbol{\Omega}^{-1}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) + \\ &\lambda(\kappa_2\|\boldsymbol{\Gamma}\boldsymbol{\beta}\|^2 + \kappa_1\|\boldsymbol{\beta}\|^2)\}\end{aligned} \tag{4}$$

where $\boldsymbol{\Omega}$ weights on each channel (dimension) of the feature vectors distinctively and $\boldsymbol{\Gamma}$ weights differently on each training sample. The last two terms alleviate the illness of Generalized Least Squares (GLS). The solution is:

$$\hat{\boldsymbol{\beta}} = \mathbf{P}\mathbf{y}, \mathbf{P} = (\mathbf{X}^T\boldsymbol{\Omega}^{-1}\mathbf{X} + \lambda(\kappa_1\mathbf{I} + \kappa_2\boldsymbol{\Gamma}^T\boldsymbol{\Gamma}))^{-1}\mathbf{X}^T\boldsymbol{\Omega}^{-1} \tag{5}$$

In order to find out an appropriate $\boldsymbol{\Gamma}$, Timofte and Van Gool [12] consider $\boldsymbol{\Omega}$ to be the identity matrix and initiate $\boldsymbol{\Gamma} = 0$. Next, for each training sample $\mathbf{x}_i$ consider

$$\hat{\boldsymbol{\beta}}_i = \mathbf{P}_i\mathbf{x}_i \tag{6}$$

where $\mathbf{P}_i$ is the reduced matrix of $\mathbf{P}$ by eliminating $\mathbf{x}_i$. Further $\hat{\boldsymbol{\beta}}_i^+$ is taken to be the vector that keeps all the squared coefficients concerning those training samples belonging to the same class of $\mathbf{x}_i$, that is, the j-th component of vector $\hat{\boldsymbol{\beta}}_i^+ = (\hat{\beta}_{i,1}^+, \hat{\beta}_{i,2}^+, \ldots, \hat{\beta}_{i,M}^+)$ satisfies:

$$\hat{\beta}_{i,j}^+ = \begin{cases} \hat{\beta}_{i,j}^2 & \text{if } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ are in the same class} \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

where $\hat{\beta}_{i,j}$ is the j-th component of $\hat{\boldsymbol{\beta}}_i$. Similarly for $\hat{\boldsymbol{\beta}}_i^- = (\hat{\beta}_{i,1}^-, \hat{\beta}_{i,2}^-, \ldots, \hat{\beta}_{i,M}^-)$ they define

$$\hat{\beta}_{i,j}^- = \begin{cases} \hat{\beta}_{i,j}^2 & \text{if } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ are not in the same class} \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

Hence, the choice of $\boldsymbol{\Gamma}$ for WCRC is:

$$\boldsymbol{\Gamma} = \text{diag}\left(\sqrt{\frac{\sum_{i=1}^M \hat{\boldsymbol{\beta}}_i^+}{\sum_{i=1}^M (\hat{\boldsymbol{\beta}}_i^+ + \hat{\boldsymbol{\beta}}_i^-)}}\right) \tag{9}$$

2.3. LCRC

In principle, our idea is a further extension of WCRC, along with a solid mathematical foundation. Motivated by the WCRC results from [12], obtained by computing $\boldsymbol{\Gamma}$ in Eq. (9), we decide to explore an optimization method with a similar strategy (*i.e*. by extracting information from the coefficients $\hat{\boldsymbol{\beta}}_i^-$). Moreover, we use fixed point theorem [1] to ensure the stability and convergence of our optimized $\boldsymbol{\Gamma}$, which was not argued in the WCRC method.

Ideally, we try to propose a method that can work as decreasing the residue $\|\mathbf{y} - \mathbf{X}_c\boldsymbol{\beta}_c\|$ when $\mathbf{y} \in$ class c. Unfortunately, it seems difficult to think this way in reality. On the other hand, if we are able to increase the residues $\{\|\mathbf{y} - \mathbf{X}_c\boldsymbol{\beta}_c\| \mid c \in \{c_1, c_2, \ldots, c_K\}, \mathbf{y} \notin \text{class } c\}$, it also can help us to improve the accuracy of recognition rate. Therefore, we optimize those $\boldsymbol{\beta}_c$ so that they tend to $\mathbf{0}$, which means query $\mathbf{y}$ becomes less dependent on training samples in different classes, while the residues become larger and tend to $\mathbf{1}$. That is the main idea behind our method.

Obviously, due to lacking of classification information we can not directly use the residues $\{\|\mathbf{y} - \mathbf{X}_c\boldsymbol{\beta}_c\| \mid c \in \{c_1, c_2, \ldots, c_K\}\}$. Hence, in our method we use a trick to deal with the problem, *e.g*. treat the training samples as 'testing samples', because we know their class labels. That is the place where $\hat{\boldsymbol{\beta}}_i^-$ comes into play.

There is another important reason underlying our assumption about training samples. Assume the training samples are large and well representative, *e.g*. for each query there always exists closed related training sample(s), then it is reasonable to replace the query with corresponding training sample in optimization. The optimization of residues concerning training samples can be approximately considered as the optimization concerning queries. If it is the case, after optimizing training samples, for a query $\mathbf{y}$ we expect the corresponding coefficients $\boldsymbol{\beta}_c$ for wrong class labels c to be very close to $\mathbf{0}$, such that they are much smaller than the coefficients concerning the training samples of same class as $\mathbf{y}$. In this way, we have a better chance to classify query y accurately. Another point is that the regulatory parameter λ helps to avoid, during the training, that for one training sample $\mathbf{x}_i$ the corresponding coefficient $\hat{\boldsymbol{\beta}}_i$ is trivial, *i.e*.

$$(\mathbf{X}^T\boldsymbol{\Omega}^{-1}\mathbf{X} + \lambda(\kappa_1\mathbf{I} + \kappa_2\boldsymbol{\Gamma}^T\boldsymbol{\Gamma}))^{-1}\mathbf{X}^T\boldsymbol{\Omega}^{-1}\mathbf{x}_i \neq \mathbf{e}_i \quad (10)$$

where $\mathbf{e}_i = (0, \ldots, 0, 1, 0, \ldots, 0)$ with 1 in i-th position.

Now we express our idea mathematically. Let

$$\boldsymbol{\gamma} = (\gamma_1, \gamma_2, \ldots, \gamma_M) \quad (11)$$

be the weights assigned to each training sample, and

$$\hat{\boldsymbol{\beta}}^-(\boldsymbol{\gamma}) = (\sum_{i=1}^{M} \hat{\boldsymbol{\beta}}_i^-(\boldsymbol{\gamma}))^2 \quad (12)$$

the accumulation of representation coefficients, where the power 2 is component-wise and used to enhance the validity of fixed point theorem, $\hat{\boldsymbol{\beta}}_i^-(\boldsymbol{\gamma})$ computed by Eqs.(6) & (8),

$$\hat{\boldsymbol{\beta}}_{i,j}^- = \begin{cases} (\mathbf{P}(\boldsymbol{\gamma})\mathbf{x}_i)_j^2 & \text{if } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ are not in the same class} \\ 0 & \text{otherwise} \end{cases} \quad (13)$$

where $\mathbf{P}(\boldsymbol{\gamma})$ [1] is the projection matrix:

$$\mathbf{P}(\boldsymbol{\gamma}) = (\mathbf{X}^T\boldsymbol{\Omega}^{-1}\mathbf{X} + \lambda(\kappa_1 I + \kappa_2 \text{diag}(\boldsymbol{\gamma})))^{-1}\mathbf{X}^T\boldsymbol{\Omega}^{-1} \quad (14)$$

Theorem 1. *Let*

$$\mathbf{f}(\boldsymbol{\gamma}) = \theta_1(\boldsymbol{\gamma} + \theta_2\hat{\boldsymbol{\beta}}^-(\boldsymbol{\gamma})) \quad (15)$$

suppose $\hat{\boldsymbol{\beta}}^-(\boldsymbol{\gamma})$ maps a closed ball O centered at zero into itself [2]*, then for $\epsilon > 0$ there exists a $\theta_2 > 0$ such that for all positive θ_1 satisfying $1 - \theta_1 > \epsilon$ it holds $\mathbf{f}(\boldsymbol{\gamma})$ admits a unique fixed point $\boldsymbol{\gamma}^*$, that is,*

$$\boldsymbol{\gamma}^* = \theta_1(\boldsymbol{\gamma}^* + \theta_2\hat{\boldsymbol{\beta}}^-(\boldsymbol{\gamma}^*)) \quad (16)$$

Proof. see the **Appendix**.

3. Learning the weights

First, we setup the coefficients for our LCRC method, then give the learning algorithm we implemented in Matlab.

3.1. Setup of $\mathbf{f}(\boldsymbol{\gamma})$

Coefficient θ_1. Our goal is to achieve $\hat{\boldsymbol{\beta}}^-(\boldsymbol{\gamma}) = 0$. For fixed point $\boldsymbol{\gamma}^*$, according to Eq. (16), $\hat{\boldsymbol{\beta}}^-(\boldsymbol{\gamma}^*)$ tends to 0 as long as θ_1 tends to 1. Hence, in our experiment we loose the condition about θ_1, simply let $\theta_1 = 1$.

Coefficient θ_2. Given $\theta_1 = 1$ and based on the proof, θ_2 should be as small as possible. However, if θ_2 is too small, then the value of $\boldsymbol{\gamma}$ keeps insignificant in the steps of optimization, so the regulatory term $\lambda\kappa_1\mathbf{I}$ in Eq. (14) will still dominate the optimization procedure. To deal with this problem, we let θ_2 be a variable depending on the steps, *i.e*. θ_2 becomes smaller and smaller with the increasing iteration steps. To this end, for n-th iteration step we let

$$\tilde{\boldsymbol{\beta}}^-(\boldsymbol{\gamma}, n) = \frac{\tilde{\theta}_2 e^{-(n/20)^2}\hat{\boldsymbol{\beta}}^-(\boldsymbol{\gamma})}{e^{2\boldsymbol{\gamma}}} \quad (17)$$

where the division is component-wise and $e^{2\boldsymbol{\gamma}}$ helps to slow down the growth rate of $\hat{\boldsymbol{\beta}}^-(\boldsymbol{\gamma})$ during the first few steps, while $e^{-(n/20)^2}$ take over the job in the next. Here, we use the term $1/20$ to weaken the dramatic decline of e^{-n^2}. For the coefficient $\tilde{\theta}_2$, after cross validation on different datasets and features discussed in the next section, empirically, for Eigenfaces it is determined to be 1 or 5, for INNLP and LDA we choose 50 and 100.

Additional term $\mathbf{l}(\boldsymbol{\gamma})$. So far, our setup of $\mathbf{f}(\boldsymbol{\gamma})$ does not directly involve the residue part $\|\mathbf{y} - \mathbf{X}_c\boldsymbol{\beta}_c\|$ of the classification decision, which is important for assisting the coefficient term to attain high recognition rate. It inspires us to introduce $\mathbf{l}(\boldsymbol{\gamma})$, whose i-th component is

$$l(\gamma)_i = \|\mathbf{x}_i - \mathbf{X}\hat{\boldsymbol{\beta}}_i^-\|^2 \quad (18)$$

so that hopefully we can optimize $\hat{\boldsymbol{\beta}}^-$ with the help of residues. However, the introduction of $\mathbf{l}(\boldsymbol{\gamma})$ may undermine the optimization procedure, after all, the main goal of our

[1] Here, we do not apply the reduced training dataset as suggested in [12] for the sake of speeding up our proposed LCRC method, and such small modification causes minor change for the performance. Besides, we use the square root of diag($\boldsymbol{\gamma}$), hence the term $\boldsymbol{\Gamma}^T\boldsymbol{\Gamma}$ in Eq. (5) becomes diag($\boldsymbol{\gamma}$) in Eq. (14).

[2] We want to assure that the invertible matrix always exists and the condition of Bananch fixed point theorem holds.

method is to minimize the value of $\hat{\boldsymbol{\beta}}^{-}$. To avoid the problem, we diminish $\mathbf{l}(\boldsymbol{\gamma})$ in the same way we handle $\hat{\boldsymbol{\beta}}^{-}(\boldsymbol{\gamma})$:

$$\tilde{\mathbf{l}}(\boldsymbol{\gamma}, n) = \frac{e^{-(n/20)^2}\mathbf{l}(\boldsymbol{\gamma})}{e^{2\boldsymbol{\gamma}}} \tag{19}$$

By imposing the coefficient $\tilde{\theta}_2$ on $\hat{\boldsymbol{\beta}}^{-}$ the optimal procedure is still dominated by $\hat{\boldsymbol{\beta}}^{-}$. Such idea is quite similar to the one of classification decision Eq. (3): residues work as the complementary term of coefficients during classification procedure. The final version of $\mathbf{f}(\boldsymbol{\gamma})$ for n-th step is

$$\mathbf{f}(\boldsymbol{\gamma}, n) = \boldsymbol{\gamma} + \tilde{\mathbf{l}}(\boldsymbol{\gamma}, n) + \tilde{\boldsymbol{\beta}}^{-}(\boldsymbol{\gamma}, n) \quad . \tag{20}$$

3.2. Algorithm

Now we present the learning algorithm in details. Algorithm (1) lists the main body of our learning weights method. Relying on the experimental data from [12] we find out that WCRC with only weighted channels (*e.g.* $\boldsymbol{\Omega}$) marginally improves the performance when compared to CRC, which has been confirmed by our own experiments as well. Therefore, we simply let $\boldsymbol{\Omega}$ to be $\mathbf{I}$ in our algorithm. Furthermore, we slightly modify λ as $\tilde{\lambda} = \text{mean}(\mathbf{X}^T\mathbf{X})\lambda$, while setting $\tilde{\kappa} = 10$ in 22-th line of Algorithm (1) in order to counterbalance the influence from $\mathbf{I}$.

Algorithm 1: LearningWeights(LCRC)

input : training samples $\mathbf{X}$
class labels $\mathbf{C}$
loop n
coefficient $\tilde{\theta}_2$
error tolerance ϵ
output: weights $\boldsymbol{\gamma}$

1 $\mathbf{P} = (\mathbf{X}^T\mathbf{X} + \tilde{\lambda}\mathbf{I})^{-1}\mathbf{X}^T$; %init. matrix $\mathbf{P}$
2 $\boldsymbol{\gamma} = 0$; %init. weights
3 $\mathbf{l} = 0$; %init. residues
4 $\hat{\boldsymbol{\beta}}^{-} = 0$; %init. coef.
5 **for** $i \leftarrow 0$ **to** n **do**
6 **for** $j \leftarrow 1$ **to** M **do**
7 **for** $k \leftarrow 1$ **to** M **do**
8 **if** $\mathbf{x}_j$ *and* $\mathbf{x}_k$ *are not in the same class* **then**
 $\hat{\boldsymbol{\beta}}^{-}_{j,k} = (\mathbf{P}(\boldsymbol{\gamma})\mathbf{x}_j)^2_k$;
9 **else**
10 $\hat{\boldsymbol{\beta}}^{-}_{j,k} = 0$;
11 **end**
12 $\hat{\boldsymbol{\beta}}^{-} = \hat{\boldsymbol{\beta}}^{-} + (\hat{\boldsymbol{\beta}}^{-}_j)^2$;
13 $l_j = \|\mathbf{x}_j - \mathbf{X}\hat{\boldsymbol{\beta}}^{-}_j\|^2$; %according to (18)
14 **end**
15 $\hat{\boldsymbol{\beta}}^{-} = (\hat{\boldsymbol{\beta}}^{-})^2$; %according to Eq. (12)
16 $\tilde{\mathbf{f}} = e^{-(i/20)^2}(\mathbf{l} + \tilde{\theta}_2\hat{\boldsymbol{\beta}}^{-})/e^{2\boldsymbol{\gamma}}$;
17 %Eq. (17), (19), (20)
18 **if** $|\tilde{\mathbf{f}}|_\infty < \epsilon$ **then**
19 break;
20 **end**
21 $\boldsymbol{\gamma} = \boldsymbol{\gamma} + \tilde{\mathbf{f}}$;
22 $\mathbf{P} = (\mathbf{X}^T\mathbf{X} + \tilde{\lambda}(\mathbf{I} + \tilde{\kappa}\,\text{diag}(\boldsymbol{\gamma}))^{-1}\mathbf{X}^T$;
23 **end**

4. Experiments

In this section, we show the results of our experiments, which can be viewed as the follow-up experiment of Timofte and Van Gool [12]. More specifically, we demonstrate how LCRC method performs concerning various datasets and dimensions. In the mean time, we also conduct a thorough investigation over the coefficient λ to find out a relatively good choice. The source codes are available from: `http://www.vision.ee.ethz.ch/~timofter/`

4.1. Setup

Datasets. We select 3 datasets AR, PIE, LFW to test the adaptability of our method. For AR dataset [4], according to [12] we have 1400 arbitrarily chosen face images for 100 individuals (14 images per person), divided into 700 training samples and 700 testing samples. When it comes to PIE face dataset, we use the subset of PIE provided by D. Cai *et al*. [2], which contains 68 individuals with near frontal poses under different illuminations and expressions, and then again randomly pick 700 training samples and 700 testing samples. As to the LFW dataset with 5749 different individuals in unconstrained condition, based on Pengfei Zhu *et al*. [17], the subset of LFW is randomly separated into 790 training- and testing samples in our experiment.

Features. Considering less efficiency and cumbersome of the original data, we do not directly test our approach on them, instead, we apply Eigenfaces [15], regularized Linear Discriminant Analysis (LDA) [3] and regularized Iterative Nearest Neighbors (INN) Linear Projections (INNLP) [10, 13] to project raw data into lower dimensions. As for the regularized Sparse Representation based Linear Projections (SRLP) [9], the experiment in [12] shows similarities to INNLP, hence we do not test it anymore in our paper. Again, as in [12] we examine our approach under various dimensions $\{5, 10, 30, 54, 99, 120, 300\}$.

Classifiers. An investigation over various CR- and none CR-based classifiers is available in [12], hence in our experiments we focus only on certain classifiers, that is, CRC, WCRC, LCRC, LCRC$_{\boldsymbol{\beta}}$ and the kernelized variants: KCRC, KWCRC, and KLCRC. LCRC$_{\boldsymbol{\beta}}$ indicates the LCRC classifier with only $\boldsymbol{\beta}$ term (no $\boldsymbol{l}$ term) in the eq. (20), line 16 of Alg. (1). Here, the kernelized classifiers use the

Gaussian kernel $k(\mathbf{x}, \mathbf{y}) = \exp(-\tau\|\mathbf{x}-\mathbf{y}\|)$, where $\tau = 0.2$ is a regulatory parameter, as in [12].

Parameters. We conduct a relatively thorough investigation over a subset of $[0.00001, 100]$ to cross-validate for λ. To this end, we randomly divide the training samples into a training proper set and a validation set with nearly same amount of samples. Hence, for AR and PIE we have 350 training and testing samples, and for LFW 390 training and 400 testing samples. We repeat such procedure and apply our approach several times (4 times in our experiments), in the end we pick up the λ with the best average performances for the final test. In Fig. 1 we plot the LCRC performance versus λ parameter in two settings. We use a logarithmic scale for the x-axis and let the collection of λ to be $\{0.0001, 0.0002, \ldots, 90, 100\}$. For both cases the λ curves grow very slowly at the beginning then decline around $\lambda = 2$. Such similar pattern confirms the effectiveness of cross validation: for dim $= 5$ the best cross-validated λ is 0.7, which corresponds to 50% recognition rate for testing data, while the maximum rate on testing data peaks at 51.0% for $\lambda = 2$. For dim $= 54$ we obtain the recognition rate 88.6% when $\lambda = 0.9$, while the maximum is 89.6% by $\lambda = 5$. The difference between the recognition rate determined by cross validation and maximum possible recognition rate is acceptable.

The behavior of $\mathbf{1}$ *and* $\hat{\boldsymbol{\beta}}^{-}$. Before we start testing LCRC approach, we need to check whether our idea works as expected, *i.e*. whether the residue $\mathbf{1}$ increases and $\hat{\boldsymbol{\beta}}^{-}$ decreases as the number of loops grows. Fig. 2 presents how the two terms, $\|\mathbf{1}\|$ and $\|\hat{\boldsymbol{\beta}}^{-}\|$, and the recognition rate change when running the optimization loop. By cross validation we set λ to be 1 for dimension 5, and 0.4 for 54, respectively (see Table (1)). The figure suggests that the value of $\|\hat{\boldsymbol{\beta}}^{-}\|$ declines gradually with the increasing steps, while the norm of residues $\|\mathbf{1}\|$ continuously increase on lower dimension if we ignore the unstable start during the first few steps. The effectiveness of LCRC is also supported by the performance of recognition rates presented by Fig. 2, the more loops the better the recognition rates for both training data and testing data, and keeping them stable in the end. We must admit that on higher dimension (54) the performance of $\|\mathbf{1}\|$ rather stay unchanged without taking the big gap at the beginning into consideration (which is consistent with the observation that normally the improvements of LCRC over WCRC is more obvious on lower dimensions than on higher dimensions), still, we succeed in optimizing the corresponding recognition rates for training and testing data. Since the recognition rate saturates after 40 iteration steps (Fig. 2), throughout the following experiments we fix the loop number n to 50. For further analyzing the adaptability of LCRC method, in the next subsection we present recognition rates for different features on AR, PIE, and LFW and the corresponding choices for λ based on cross validation.

4.2. Performance of classifiers

AR dataset. Table 1 indicates that LCRC has achieved progress over CRC and WCRC on dataset AR. First of all, if we only optimize the $\boldsymbol{\beta}$ term of LCRC, *i.e*. by applying LCRC$_{\boldsymbol{\beta}}$, it has demonstrated nearly as competitive performance as WCRC, which can be confirmed by the performances on INNLP and LDA. Together with the complementary residue term we obtain rather an impressive performance with LCRC, especially for lower dimensions (< 99). For example, for 5-dimensional INNLP projections the recognition rate increases more than 8% over WCRC. More interestingly, if we focus on projections like INNLP and LDA, the recognition rate obtained by LCRC shows convincing evidence that it is nearly competitive to kernel based methods or query-adapted methods such as adaptive WCRC (AWCRC) [12], which extract extra information from testing samples. For AWCRC please check the experimental results from [12].

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

Features	Method	5	10	30	54	99	120	300
INNLP	CRC	11.3	28.6	76.8	89.4	93.6	94.3	94.0
	WCRC	26.2	56.7	86.6	90.6	92.4	94.3	94.3
	LCRC$_{\boldsymbol{\beta}}$	26.2	52.9	82.6	90.7	**93.9**	93.9	93.7
	LCRC	**34.2**	**60.7**	**87.3**	**91.9**	92.9	**94.4**	**94.6**
	KCRC	34.1	59.2	84.3	92.0	94.1	94.7	95.0
	KWCRC	29.3	59.7	**86.7**	92.3	94.0	**94.9**	**95.1**
	KLCRC	**34.6**	**63.1**	86.4	**93.1**	**94.6**	94.3	94.4
	λ_{CRC}	20	1	0.2	0.1	1	0.7	0.2
	λ_{WCRC}	5	0.9	0.5	0.5	0.5	0.07	0.03
	$\lambda_{\text{LCRC}_{\boldsymbol{\beta}}}$	0.08	0.02	0.2	0.6	2	0.04	0.04
	λ_{LCRC}	1	0.6	0.6	0.4	0.7	0.09	0.02
	λ_{KCRC}	0.1	0.1	0.3	1	1	1	0.5
	λ_{KWCRC}	0.1	0.1	1	1	1	0.2	0.1
	λ_{KLCRC}	0.1	1	0.1	0.3	1	1	0.9
LDA	CRC	15.5	37.9	80.5	90.8	94.1		
	WCRC	27.6	54.7	86.7	**92.4**	**94.3**		
	LCRC$_{\boldsymbol{\beta}}$	30.3	57.8	85.1	91.1	92.1		
	LCRC	**34.1**	**58.7**	**88.3**	91.9	94.1		
	KCRC	31.2	58.5	85.3	91.6	94.4		
	KWCRC	29.6	56.4	**87.6**	**92.7**	94.4		
	KLCRC	**33.9**	**62.4**	86.7	92.3	**95.0**		
	λ_{CRC}	6	0.04	0.02	0.007	0.7		
	λ_{WCRC}	0.3	0.01	0.02	0.06	0.1		
	$\lambda_{\text{LCRC}_{\boldsymbol{\beta}}}$	0.3	0.001	0.001	0.002	0.001		
	λ_{LCRC}	0.7	0.01	0.2	0.0003	0.06		
	λ_{KCRC}	0.1	0.1	0.5	0.5	1		
	λ_{KWCRC}	0.1	0.1	0.1	0.1	0.1		
	λ_{KLCRC}	0.1	0.1	0.1	0.1	0.3		
Eigenfaces	CRC	07.0	19.5	64.4	80.5	89.1	90.4	**94.0**
	WCRC	09.2	32.5	72.7	83.7	89.1	90.4	93.7
	LCRC	**10.9**	**36.3**	**74.8**	**84.1**	**90.3**	**90.6**	93.7
	KCRC	18.3	44.6	77.4	**83.6**	87.6	**88.8**	91.0
	KWCRC	21.3	47.8	77.5	83.4	**88.1**	88.7	**91.1**
	KLCRC	**24.0**	**47.9**	**77.7**	83.0	87.4	88.4	**91.1**
	λ_{CRC}	2	0.02	0.004	0.008	0.0004	0.01	0.009
	λ_{WCRC}	0.7	0.04	0.0003	0.003	0.0004	0.007	0.0008
	λ_{LCRC}	0.06	0.005	0.0004	0.002	0.0008	0.004	0.001
	λ_{KCRC}	0.0001	0.0001	0.003	0.006	0.01	0.02	0.006
	λ_{KWCRC}	0.0001	0.0001	0.0001	0.0006	0.003	0.003	0.0006
	λ_{KLCRC}	0.0003	0.0001	0.0001	0.004	0.0007	0.0005	0.0004

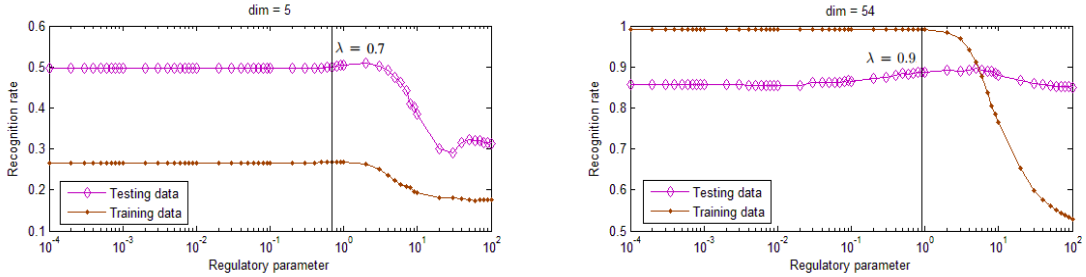

Figure 1. LCRC performance vs. regulatory parameter on PIE dataset with LDA projections.

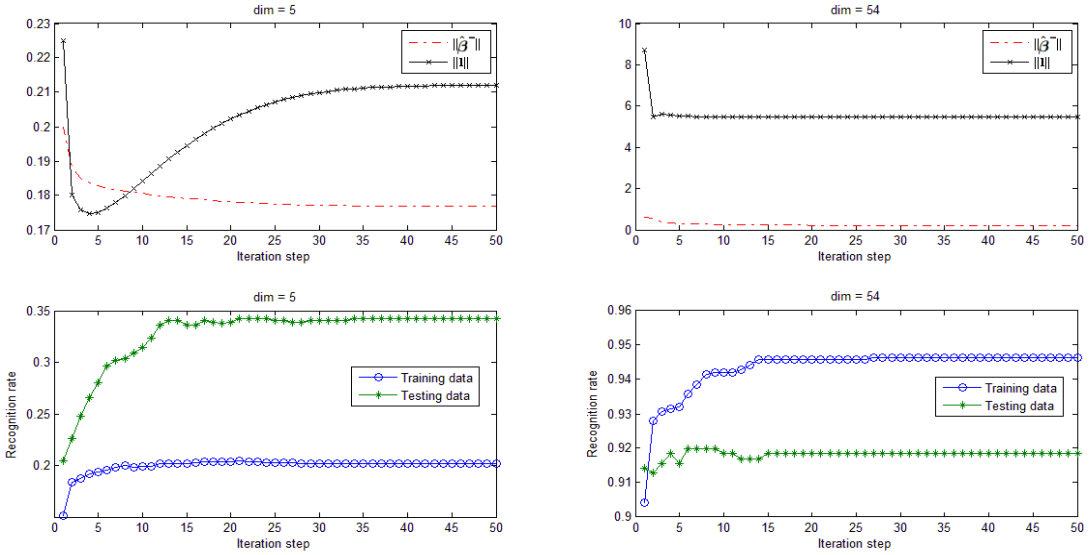

Figure 2. LCRC residues, coefficients energies, and recognition rates vs. iteration step on AR dataset with INNLP projections.

PIE dataset. Table 2 demonstrates similar performance pattern for LCRC on PIE dataset as on AR dataset. For instance, it shows continuous improvements over both WCRC and KCRC involving INNLP and LDA projections. Again, for INNLP and LDA our results of LCRC, a query-independent method, are relatively competitive to KWCRC or query-adapted classifiers from [12], which was not accomplished by WCRC. On the other hand, on higher dimensions ($\geq$ 99) the classifiers, other than CRC, do not present enough improvements. We think that, given CRC has already reached quite impressive recognition rate on higher dimensions, there may not be much room left for extra improvement, hence we do not recognize obvious progress made by WCRC/LCRC. We note that (K)LCRC barely optimize the results of (K)WCRC when it comes to the Eigenface features . It is believed that the modified training data may lose discriminative properties during the procedure and become uniform, which makes LCRC less effective.

LFW dataset. The recognition rates presented in Table 3 offer us a different point of view about how LCRC performs under unconstrained conditions – the face images are not strongly aligned as in AR or PIE datasets. The results consistently substantiate the improvement made by LCRC over WCRC or CRC. More importantly, even on higher dimensions ($\geq$ 99) LCRC continuously demonstrates relatively obvious progress. For example, on 99-dimensional LDA projections LCRC increases the recognition rates more than 7% over CRC, and 6% over WCRC. This indicates LCRC method indeed alleviates the illness of CRC for less well-performed cases, and also reveals the optimization potential when the current CRC method is not accurate enough.

Raw samples. In the previous cases, we first reduce the dimensionality of the data then apply the classifiers. Hence, it is natural to ask how does LCRC perform on raw samples? Fig. 3 depicts the performances for CRC, WCRC, and LCRC on AR dataset. Again, we use cross validation to determine the best λ for distinct methods and dimensions. Next we record the results from 30 to 2580 gray pixels. As it turned out, LCRC shows mild improvements over WCRC on raw samples, which is in keeping with our expectations.

Table 2. Face recognition rates [%] on PIE.

Features	Method	5	10	30	54	99	120	300
INNLP	CRC	22.1	42.1	76.6	86.0	88.1	88.4	88.6
	WCRC	42.1	72.0	84.6	86.6	88.6	88.3	**88.7**
	LCRC	**50.7**	**73.6**	**86.1**	**88.3**	**88.7**	**88.6**	**88.7**
	KCRC	50.9	73.4	87.1	89.0	88.4	89.1	89.4
	KWCRC	52.0	73.6	**87.7**	89.6	**90.0**	**89.9**	89.4
	KLCRC	**55.0**	**74.6**	87.3	**90.0**	89.3	89.7	**90.3**
	λ_{CRC}	60	10	0.3	4	6	6	1
	λ_{WCRC}	0.6	3	2	2	1	1	0.1
	λ_{LCRC}	1	0.7	1	0.7	0.7	0.5	0.08
	λ_{KCRC}	0.1	0.1	1	1	1	1	1
	λ_{KWCRC}	0.1	0.1	1	1	1	1	0.1
	λ_{KLCRC}	0.1	0.1	0.3	1	1	1	0.7
LDA	CRC	31.1	60.6	82.4	86.6			
	WCRC	44.9	74.0	85.6	86.9			
	LCRC	**50.0**	**74.0**	**87.4**	**88.6**			
	KCRC	49.7	70.0	85.3	88.1			
	KWCRC	51.4	**73.4**	86.6	88.3			
	KLCRC	**53.3**	72.6	**87.1**	**89.0**			
	λ_{CRC}	0.4	0.1	0.09	1			
	λ_{WCRC}	0.4	0.07	0.8	0.2			
	λ_{LCRC}	0.7	0.04	0.6	0.9			
	λ_{KCRC}	0.1	0.1	0.3	2			
	λ_{KWCRC}	0.1	0.1	0.1	0.2			
	λ_{KLCRC}	0.1	0.1	0.1	0.4			
Eigenfaces	CRC	02.7	15.6	50.4	70.0	79.7	83.1	87.1
	WCRC	**05.7**	28.0	**68.3**	**76.4**	**83.4**	**84.3**	**87.4**
	LCRC	05.4	**30.9**	67.0	75.9	82.0	83.9	87.3
	KCRC	09.3	34.6	67.1	75.0	80.1	80.4	**83.4**
	KWCRC	09.1	**36.6**	**67.6**	75.3	80.1	80.4	**83.4**
	KLCRC	**10.6**	36.3	67.3	**76.0**	**80.3**	**81.1**	83.0
	λ_{CRC}	1	0.5	0.008	0.009	0.002	0.009	0.01
	λ_{WCRC}	0.07	0.003	0.007	0.006	0.004	0.001	0.002
	λ_{LCRC}	0.2	0.05	0.03	0.003	0.0009	0.002	0.0009
	λ_{KCRC}	0.00001	0.0003	0.003	0.007	0.008	0.01	0.006
	λ_{KWCRC}	0.00002	0.00008	0.0003	0.0007	0.0005	0.0009	0.0003
	λ_{KLCRC}	0.00003	0.0003	0.0002	0.0004	0.0005	0.0003	0.0007

Table 3. Face recognition rates [%] on LFW.

Features	Method	5	10	30	54	99	120	300
INNLP	CRC	04.3	11.0	22.4	28.9	38.5	37.1	42.0
	WCRC	06.8	**19.5**	32.0	33.0	39.9	40.1	43.3
	LCRC	**08.5**	**19.5**	**33.4**	**38.0**	**44.8**	**44.1**	**44.2**
	KCRC	07.5	17.7	27.2	33.5	41.4	42.8	42.4
	KWCRC	07.9	18.9	**33.3**	**35.4**	43.2	43.2	43.4
	KLCRC	**08.7**	**19.8**	31.0	34.8	**43.5**	**45.8**	**45.6**
	λ_{CRC}	20	20	0.01	0.001	0.3	0.02	0.5
	λ_{WCRC}	3	0.003	0.7	0.9	0.2	0.1	0.08
	λ_{LCRC}	0.6	1	0.4	0.4	0.4	0.2	0.06
	λ_{KCRC}	0.01	0.01	0.01	0.03	1	1	0.5
	λ_{KWCRC}	0.01	0.01	0.1	0.3	0.3	0.04	0.09
	λ_{KLCRC}	0.01	0.01	0.01	0.01	5	2	0.4
LDA	CRC	05.3	13.7	24.9	31.3	37.5	36.7	
	WCRC	07.2	18.9	30.0	**33.4**	38.5	40.3	
	LCRC	**08.5**	**19.9**	**32.5**	33.2	**44.6**	**44.1**	
	KCRC	06.7	18.4	25.1	30.3	38.7	41.1	
	KWCRC	**08.2**	18.1	30.4	34.2	**41.1**	41.4	
	KLCRC	**08.2**	**19.4**	**30.9**	**35.3**	40.3	**41.5**	
	λ_{CRC}	1	0.04	0.02	0.02	0.01	0.002	
	λ_{WCRC}	0.2	0.003	0.01	0.1	0.02	0.05	
	λ_{LCRC}	0.6	0.02	0.3	0.0001	0.3	0.2	
	λ_{KCRC}	0.01	0.01	0.01	0.02	0.08	0.2	
	λ_{KWCRC}	0.01	0.01	0.01	0.01	0.02	0.02	
	λ_{KLCRC}	0.01	0.01	0.01	0.01	0.01	0.01	
Eigenfaces	CRC	02.2	05.1	17.3	23.6	30.3	32.7	40.8
	WCRC	**02.7**	**06.8**	22.5	**30.9**	**36.7**	38.5	42.9
	LCRC	02.2	**06.8**	**22.9**	30.5	36.6	**38.7**	**43.4**
	KCRC	03.5	07.9	21.8	26.1	31.3	31.5	36.8
	KWCRC	03.9	**08.6**	20.6	26.3	32.2	33.3	37.6
	KLCRC	**04.2**	08.4	**22.3**	**28.5**	**33.2**	**34.1**	**38.1**
	λ_{CRC}	6	0.05	0.05	0.2	0.07	0.04	0.04
	λ_{WCRC}	1	0.2	0.08	0.08	0.03	0.02	0.007
	λ_{LCRC}	0.1	0.3	0.02	0.04	0.01	0.02	0.006
	λ_{KCRC}	0.02	0.02	0.01	0.01	0.09	0.02	0.02
	λ_{KWCRC}	0.1	0.09	0.01	0.01	0.01	0.01	0.01
	λ_{KLCRC}	0.03	0.04	0.01	0.2	0.06	0.06	0.01

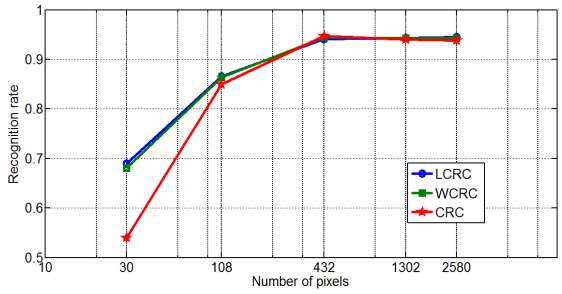

Figure 3. CRC, WCRC, and LCRC performances on AR dataset with raw gray scale pixels from scaled images.

4.3. Running time

Considering LCRC is an extension of WCRC by introducing additional optimization steps, the LCRC training time is dependent on the optimization steps and more specifically, it increases linearly with the growth of iteration steps. Still learning the weights is considered to be inexpensive, in the order of seconds for our experimental settings. In Table 4 we report the running time on CRC, WCRC and LCRC with Intel Core i7-2600K(3.40GHz). We give the training time and testing time on AR datasets with LDA features, which confirms the time complexity we discussed. Since LCRC does not exploit information from testing samples, we can precompute the projection matrix $\mathbf{P}$ offline at training time then apply it. Therefore, LCRC shares the same time complexity at test with CRC and WCRC, which is significantly lower than the query-adapted methods (such as SRC or AWCRC, as shown in [12]). For real world applications the running time is a crucial factor, especially at test.

Table 4. Total running time [s] on AR with LDA features.

Phase	Dimensionality	CRC	WCRC	LCRC
Training (offline)	5	0.0056	0.0441	1.3811
	10	0.0057	0.0495	1.9548
	30	0.0066	0.0610	2.1402
	54	0.0071	0.0679	2.9173
	99	0.0080	0.0862	3.7151
Testing (online)	5	0.7155	0.7200	0.7184
	10	0.7367	0.7299	0.7319
	30	0.8268	0.8171	0.8282
	54	0.9359	0.9300	0.9351
	99	0.9325	0.9352	0.9356

5. Conclusions

As the subsequent part of Timofte and Van Gool [12], we answer the question how to learn the (optimal) weights. We give a simple and rigorous proof not only to verify our optimization idea, but also to reason why the weights used in WCRC shows improvements, mainly because the choice

in WCRC is the starting point towards optimal weights by our new method. We employ different strategies to show that LCRC has a wide applicability without over-fitting specific datasets. In the mean time, though by introducing optimization steps LCRC costs more computational time than WCRC, it keeps the computation offline, which is a great advantage over query-adapted methods like AWCRC or SRC. Last but not least, LCRC approach implies that we can cure the 'wrong distance' between samples by imposing optimized matrix, this leaves us the clue that the given samples may not lie in a flat space, rather they are in a curved space (manifold) which has own intrinsic geometric structure. So the future challenge for us is to train an appropriate manifold which can be used to better describe the 'true distance' between samples.

Acknowledgments This was partly supported by the ETH General Founding (OK) and the European Research Council (ERC) under the project VarCity (#273940).

Appendix

***Proof.*(Theorem 1)** Since $\hat{\boldsymbol{\beta}}^{-}(\boldsymbol{\gamma})$ maps a closed ball O centered at zero into itself, it implies that

$$\mathbf{Q}(\boldsymbol{\gamma}) = (\mathbf{X}^T\boldsymbol{\Omega}^{-1}\mathbf{X} + \lambda(\kappa_1\mathbf{I} + \kappa_2\text{diag}(\boldsymbol{\gamma}))) \tag{21}$$

is invertible on O. Once $\mathbf{Q}(\boldsymbol{\gamma})$ is invertible, the partial derivative always exists

$$\frac{\partial\mathbf{Q}(\boldsymbol{\gamma})^{-1}}{\partial\gamma_i} = -\mathbf{Q}(\boldsymbol{\gamma})^{-1}\frac{\partial\mathbf{Q}(\boldsymbol{\gamma})}{\partial\gamma_i}\mathbf{Q}(\boldsymbol{\gamma})^{-1} \tag{22}$$

for $i \in \{1, 2, \ldots, M\}$, and together with $\mathbf{Q}(\boldsymbol{\gamma})$ differentiable on O, we conclude that $\hat{\boldsymbol{\beta}}^{-}(\boldsymbol{\gamma})$ is differentiable on O, furthermore, $\hat{\boldsymbol{\beta}}^{-}(\boldsymbol{\gamma})$ is Lipschitz continuous, *i.e*.

$$\exists k > 0 : \|\hat{\boldsymbol{\beta}}^{-}(\boldsymbol{\gamma}_1) - \hat{\boldsymbol{\beta}}^{-}(\boldsymbol{\gamma}_2)\| \leq k\|\boldsymbol{\gamma}_1 - \boldsymbol{\gamma}_2\|. \tag{23}$$

Then by triangular inequality and (23) it holds:

$$\begin{aligned}\|\mathbf{f}(\boldsymbol{\gamma}_1) - \mathbf{f}(\boldsymbol{\gamma}_2)\| &= \theta_1\|\boldsymbol{\gamma}_1 - \boldsymbol{\gamma}_2 + \theta_2(\hat{\boldsymbol{\beta}}^{-}(\boldsymbol{\gamma}_1) - \hat{\boldsymbol{\beta}}^{-}(\boldsymbol{\gamma}_2))\| \\ &\leq (1-\epsilon)(1+\theta_2 k)\|\boldsymbol{\gamma}_1 - \boldsymbol{\gamma}_2\|\end{aligned} \tag{24}$$

So it is not difficult to find a sufficiently small $\theta_{2,k,\epsilon}$ (dependent on k, ϵ) so that it satisfies

$$(1-\epsilon)(1+\theta_{2,k,\epsilon}k) < 1 \tag{25}$$

hence $\mathbf{f}(\boldsymbol{\gamma})$ is a contraction.

For applying the Banach fixed point theorem [1], we still need to make sure that $\mathbf{f}(\boldsymbol{\gamma})$ maps a closed ball to itself. This can be achieved by picking up a sufficient small $c'_{2,k,\epsilon}$. Finally, we set

$$\theta_2 = \min\{\theta'_{2,k,\epsilon}, \theta_{2,k,\epsilon}\}. \tag{26}$$

Then we guarantee that we can apply fixed point theorem for each ϵ, and it implies in theory, we are allowed to pick ϵ sufficiently close to 1 and for the corresponding fixed point $\boldsymbol{\gamma}$ it holds $\hat{\boldsymbol{\beta}}^{-}(\boldsymbol{\gamma}) \approx 0$.

References

[1] S. Banach. Sur les opérations dans les ensembles abstraits et leur application aux équations intégrales. *Fund. Math.*, 3:133–181, 1922.

[2] D. Cai, X. He, and J. Han. Efficient kernel discriminant analysis via spectral regression. In *Proc. Int. Conf. on Data Mining (ICDM'07)*, 2007.

[3] R. Fisher. The statistical utilization of multiple measurements. *Annals of Eugenics*, 8:376–386, 1938.

[4] A. Martinez and R. Benavente. The AR face database. Technical report, CVC Tech, 1998.

[5] C. Saunders, A. Gammerman, and V. Vovk. Ridge regression learning algorithm in dual variables. In *ICML*, pages 515–521, 1998.

[6] B. Schölkopf, J. C. Platt, and T. Hoffman. Statistical analysis with missing data, second ed. In *NIPS*, pages 801 – 808. MIT Press, 2002.

[7] J. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural Processing Letters*, 9(3):293 – 300, 1999.

[8] A. Tikhonov and V. Arsenin. Solution of ill-posed problems. *Winston & Sons*, 1977.

[9] R. Timofte and L. Van Gool. Sparse representation based projections. In *BMVC*, 2011.

[10] R. Timofte and L. Van Gool. Iterative nearest neighbors for classification and dimensionality reduction. In *CVPR*, pages 2456–2463, 2012.

[11] R. Timofte and L. Van Gool. Weighted collaborative representation and classification of images. In *ICPR*, pages 1606–1610, 2012.

[12] R. Timofte and L. Van Gool. Adaptive and weighted collaborative representations for image classification. *Pattern Recognition Letters*, 43, 2014.

[13] R. Timofte and L. Van Gool. Iterative nearest neighbors. *Pattern Recognition*, 48:60–72, 2015.

[14] J. Wright, A. Y. Yang, A. Ganesh, S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Learning*, 31(2), February 2009.

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

[16] L. Zhang, W. Zhou, P. Chang, J. Liu, Z. Yan, T. Wang, and F. Li. Kernel sparse representation-based classifier. *IEEE Transactions on Signal Processing*, 60(4):1684 –1695, 2012.

[17] P. Zhu, L. Zhang, Q. Hu, and S. C. K. Shiu. Multi-scale patch based collaborative representation for face recognition with margin distribution optimization. *ECCV*, 7572:822–835, 2012.

[18] H. Zou, T. Hastie, and R. Tibshirani. Sparse principal component analysis. *JCGS*, 15:262–286, 2006.