Self-explanatory Sparse Representation for Image Classification

Bao-Di Liu[1,⋆], Yu-Xiong Wang[2,⋆], Bin Shen[3,⋆⋆], Yu-Jin Zhang[4], and Martial Hebert[2]

[1] Col. of Information and Control Engineering, China University of Petroleum, Qingdao 266580, China
thu.liubaodi@gmail.com
[2] Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA
yuxiongw@cs.cmu.edu, hebert@ri.cmu.edu
[3] Dept. of Computer Science, Purdue University, West Lafayette, IN 47907, USA
bshen@purdue.edu
[4] Dept. of Electronic Engineering, Tsinghua University, Beijing 100084, China
zhang-yj@mail.tsinghua.edu.cn

Abstract. Traditional sparse representation algorithms usually operate in a single Euclidean space. This paper leverages a self-explanatory reformulation of sparse representation, i.e., linking the learned dictionary atoms with the original feature spaces explicitly, to extend simultaneous dictionary learning and sparse coding into reproducing kernel Hilbert spaces (RKHS). The resulting single-view self-explanatory sparse representation (SSSR) is applicable to an arbitrary kernel space and has the nice property that the derivatives with respect to parameters of the coding are independent of the chosen kernel. With SSSR, multiple-view self-explanatory sparse representation (MSSR) is proposed to capture and combine various salient regions and structures from different kernel spaces. This is equivalent to learning a nonlinear structured dictionary, whose complexity is reduced by learning a set of smaller dictionary blocks via SSSR. SSSR and MSSR are then incorporated into a spatial pyramid matching framework and developed for image classification. Extensive experimental results on four benchmark datasets, including UIUC-Sports, Scene 15, Caltech-101, and Caltech-256, demonstrate the effectiveness of our proposed algorithm.

Keywords: Reproducing Kernel Hilbert Spaces, Sparse Representation, Multiple View, Image Classification.

1 Introduction

After decades of effort, the power of sparse representation has been gradually revealed in visual computation areas, such as image annotation [21,22], image

⋆ Contributed equally to this paper.
⋆⋆ Corresponding author.

D. Fleet et al. (Eds.): ECCV 2014, Part II, LNCS 8690, pp. 600–616, 2014.

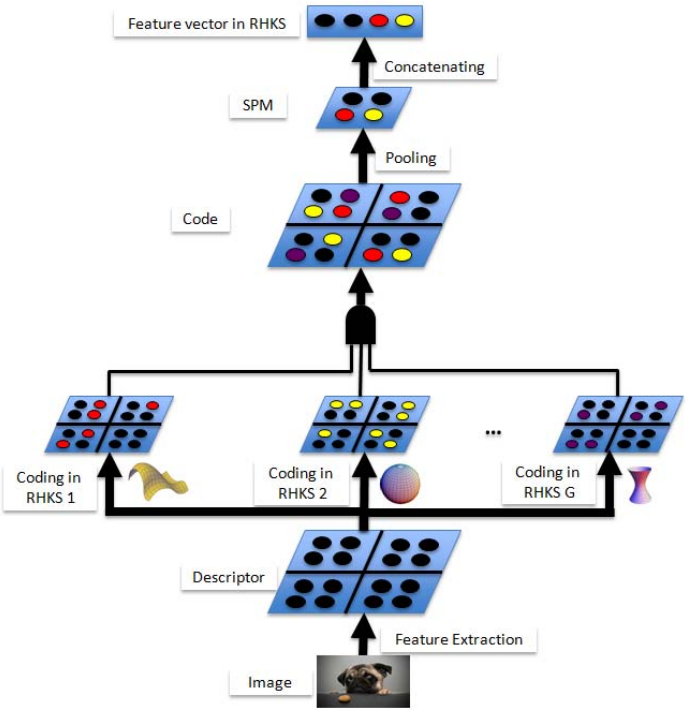


Fig. 1. Flowchart of the proposed SSSR & MSSR dictionary learning and coding process for image classification

inpainting [27,36], and have achieved impressive performance. Different from traditional decomposition frameworks like PCA, non-negative matrix factorization [28,29,37,39], and low-rank factorization [30,31,35], sparse representation [16] allows coding under over-complete bases (i.e., the number of bases is greater than the input data dimension), and thus generates sparse codes capable of representing the data more adaptively.

One example task is image classification [18,19,40], which aims to associate images with semantic labels automatically. The most common framework is the discriminative model [12,38,40]. There are five main steps: feature extraction, dictionary learning, image coding, image pooling, and SVM-based classification. Dictionary (also called vocabulary) learning is the key step here. One standard version of vocabulary learning is K-means clustering on image patches combined with hard- or soft-assignment vector quantization (VQ) [7]. Spatial pyramid matching (SPM) is typically incorporated in the pipeline to compensate the loss of spatial information [12]. In 2009, [40] introduced sparse representation algorithm for learning dictionary and coding images based on SPM, resulting in state-of-the-art performance in image classification.

Works of this kind usually operate in the original Euclidean space, which cannot capture nonlinear structures hidden in the data. Meanwhile, image descriptors often have intrinsic nonlinear similarity measures, such as SPM. A classical way to deal with this is to adopt the "*kernel trick*" [25], which maps the features into high dimensional feature space to make features of different categories more linearly separable. In this case, we may find a sparse representation for the features more easily [5,41]. With the introduction of kernel techniques, the learned dictionary becomes versatile. For the K-means based scheme, [38] learned a dictionary in the histogram intersection kernel (HIK) space, while [8] learned it in the Gaussian radial basis function (RBF) kernel space. For the sparse representation based scheme, [23] proposed kernel K-SVD and kernel MOD methods. [4] proposed kernel sparse representation (KSR), where the dictionary is trimmed to work well with a simplified Gaussian Mixture Model which can be viewed as a solution to density estimation problems. It generally outperforms the previous alternative extensions of sparse representation for image classification and face recognition. However, applications are very restricted since the derivation is exclusively based on the property of RBF kernel. That is, this method is limited to a few specific kernels and there are many useful kernels for which even the kernel functions cannot be expressed mathematically. To cover arbitrary kernel spaces, their other work [5] instead learned the dictionary first in the original space, and then mapped it to the high dimensional ambient space, whose improved performance was shown by using HIK. Unfortunately, this procedure is only an approximation and does not solve exact dictionary learning in the kernel space. [33] aims to make kernel-based classifiers efficient in both space and time. Sparse coding here is exploited to approximate the mapped features in the kernel space. There is no dictionary learning involved. They apply it to large image feature vectors, such as Fisher encoding, and the cost of non-linear SVM prediction is reduced by this approximation while maintaining the classification accuracy above an acceptable threshold. [3,20] point out that the data are self-explanatory. These approaches have many applications. However, treating the samples as a dictionary is almost impossible for the application of image classification based on bags of words, since we may easily have more than millions of local features to form the matrix. Thus, it is usually too expensive to calculate the sparse codes. [17] proposed a self-explanatory convex sparse representation for image classification. However, the additional convexity constraint is too restrictive to obtain better performance in practice.

Given that existing work either handles specific kernels or is implemented as an approximation, an issue arises naturally: we need a systematic scheme to generalize sparse representation into reproducing kernel Hilbert spaces (RKHS), which can directly learn dictionaries for arbitrary kernels. This leads to the first contribution of this paper, i.e., single-view self-explanatory sparse representation (SSSR). The key idea here is a new formulation as self-explanatory sparse representation inspired by the representer theorem of Schölkopf et al. [26], which enforces each dictionary atom in the RKHS to lie in the span of the features. Owing to the properties of this reformulation, the sparse representation can be

tractably solved in arbitrary kernel spaces. The procedures do not require that the mathematical form of any kernel function be known, rather they work directly on the kernel matrices. It thus has the nice property that the derivatives with respect to parameters of the coding are independent of the used kernel. It also presents an explicit relationship between the basis vectors and the original image features, leading to enhanced interpretability.

On the other hand, a single kernel is generally not enough. Multiple kernel learning (MKL) and multiple view learning have thus been flourishing in computer vision [9]. Different kernels correspond to different implicit feature transformations, which result in different measures of similarity in the original feature space. MKL tries to integrate the power of different kernels by learning a weighted linear combination of them. A typical example is [42], which selects different input features and combines them by mapping them to a homogeneous Gaussian RBF kernel space.

Motivated by the success of the above SSSR for arbitrary kernels and multiple kernel learning [32], we propose multiple-view self-explanatory sparse representation (MSSR) to identify and combine various salient regions and structures from different kernel spaces. This is the second contribution of this paper. It is equivalent to learning a dictionary with non-linear structure, whose complexity is reduced by learning a set of smaller dictionary blocks via SSSR. Slightly different from the typical MKL scenario metioned above [42], here we exploit the nonlinear representation capability. That is, only a single source of the original image features is chosen while various kernel subspaces are merged. To effectively solve the corresponding sparse coding subproblem and dictionary learning subproblem, feature-sign search [13] and Lagrange multipliers are then generalized in the high dimensional space. As an application example, we incorporate SSSR and MSSR into the spatial pyramid matching framework and develop them for image classification. In fact, SSSR and MSSR could also be used in many other applications. The extensive experimental results demonstrate that the proposed SSSR and MSSR algorithms can learn more discriminative sparse codes than sparse coding, leading to improved performance in image classification. A flowchart of the proposed algorithm is illustrated in Figure 1.

The rest of the paper is organized as follows. Section 2 overviews sparse representation briefly, and introduces self-explanatory sparse representation reformulation naturally. SSSR and MSSR algorithms are proposed in Section 3. The solutions to the corresponding optimization problems are elaborated in Section 4. The overall algorithm is also summarized. Experimental results on several benchmark datasets are given in Section 5. Finally, discussions and conclusions are drawn in Section 6.

2 Self-explanatory Sparse Representation

We assume that the data vectors can be represented as linear combinations of only few active basis vectors that carry the majority of the energy of the data.

Formally, we solve the following problem:

$$\begin{aligned} \min_{\boldsymbol{B},\boldsymbol{S}} f(\boldsymbol{B},\boldsymbol{S}) &= \|\boldsymbol{X}-\boldsymbol{B}\boldsymbol{S}\|_F^2 + 2\alpha\|\boldsymbol{S}\|_1 \\ s.t.\ \ &\|\boldsymbol{B}_{\bullet i}\|_2 \leq 1, \forall i = 1,2,\ldots,K. \end{aligned} \tag{1}$$

Here, $\boldsymbol{X} \in \mathbb{R}^{D\times N}$ represents the local descriptors extracted from images, where D is the dimension of $\boldsymbol{X}$, and N is the number of samples in $\boldsymbol{X}$. $\boldsymbol{B} \in \mathbb{R}^{D\times K}$ is the dictionary, where K is the size of the dictionary. $\boldsymbol{S} \in \mathbb{R}^{K\times N}$ is the corresponding sparse codes. $\|\cdot\|_F^2$ represents the Frobenius norm. $\boldsymbol{B}_{\bullet i}$ and $\boldsymbol{B}_{j\bullet}$ denote the i-th column and j-th row vectors of matrix $\boldsymbol{B}$, respectively. The regularization term is to control sparsity in $\boldsymbol{S}$, where α is a regularization parameter balancing the tradeoff between fitting goodness and sparseness.

However, there is no explicit relationship between the learned dictionary and the original features in the above formulation. Notice that K-means can be viewed as a special case of sparse representation with $\|\boldsymbol{S}_{\bullet i}\|_0 = 1, \|\boldsymbol{S}_{\bullet i}\|_1 = 1, \boldsymbol{S}_{\bullet i} \geq 0$, while its learned dictionary atoms are the centroids of the input data. Hence, for reasons of interpretability it may be useful to impose the constraint that each basis vector lies within the column space of the original features $\boldsymbol{X}$. By introducing the weight matrix $\boldsymbol{W} \in \mathbb{R}^{N\times K}$ and substituting the bases $\boldsymbol{B}$ in (1) with $\boldsymbol{X}\boldsymbol{W}$, we get a new formulation as self-explanatory sparse representation:

$$\begin{aligned} \min_{\boldsymbol{W},\boldsymbol{S}} f(\boldsymbol{W},\boldsymbol{S}) &= \|\boldsymbol{X}-\boldsymbol{X}\boldsymbol{W}\boldsymbol{S}\|_F^2 + 2\alpha\|\boldsymbol{S}\|_1 \\ s.t.\ \ &\|\boldsymbol{X}\boldsymbol{W}_{\bullet k}\|_2 \leq 1, \forall k = 1,2,\ldots,K. \end{aligned} \tag{2}$$

Typically $K \ll N$, and the trivial solution $\boldsymbol{W} = \boldsymbol{I}$ is thus naturally ruled out. Actually, these two formulations can be unified from the perspective of the representer theorem of Schölkopf et al. [26]. When applied to the linear kernel, the solution to Eqn. 1, when minimizing over $\boldsymbol{B}$, is going to be of the form $\boldsymbol{B} = \boldsymbol{X}\boldsymbol{W}$. Hence, Eqn. 2 can be intuitively viewed as the "dual" reformulation of Eqn. 1, the "primal" form of the sparse representation problem, and gives the same solution. This is better understood if one draws an analogy with linear SVM training: one can formulate the training problem as an optimization over 1) either directly the weights of a linear classifier vector of the same dimension as the input signal 2) or the support vectors weights, that is a vector of dimension equal to the size of the training set that is used to linearly combine the training inputs. In the context of the problem here, the linear classifier is analogous to the dictionary $\boldsymbol{B}$, and the support vector weights are analogous to the weights $\boldsymbol{W}$.

Replacing the bases with linear combinations of image features has several advantages. The atoms now capture a notion of centroids similar to K-means, which explicitly expresses what happens during dictionary learning, leading to enhanced interpretability. Correspondingly, the code $\boldsymbol{S}$ can be interpreted as the posterior cluster probabilities and the weight $\boldsymbol{W}$ can be considered as the contributions of each data point when learning bases. Sparse representation and K-means can be thus unified in the same framework. Moreover, by confining the search space of potential bases, it might limit overfitting. The weight $\boldsymbol{W}$ makes

the scenario more flexible, and different constraints like non-negativity can be incorporated into it so as to adapt to various tasks, while they might be difficult to directly impose on $\boldsymbol{B}$. An obvious cost of the reformulation is the increased computational complexity, because $D \ll N$ generally for over-complete representation. However, we will soon discover in the next section that it actually facilitates our solution with executable steps in the nonlinear kernel spaces.

3 Single- and Multiple-View Self-explanatory Sparse Representation

3.1 Single View Formulation

Besides the interpretability, another important property for self-explanatory sparse representation is that it is easy to kernelize due to the separation of original data. Suppose that there exists a feature mapping function $\phi : \mathbb{R}^D \to \mathbb{R}^t$. It maps the original feature space to the high dimensional kernel space: $\boldsymbol{X} = [\boldsymbol{X}_{\bullet 1}, \boldsymbol{X}_{\bullet 2}, \cdots, \boldsymbol{X}_{\bullet N}] \to \phi(\boldsymbol{X}) = [\phi(\boldsymbol{X}_{\bullet 1}), \phi(\boldsymbol{X}_{\bullet 2}), \cdots, \phi(\boldsymbol{X}_{\bullet N})]$. Then, the objective function of (2) can be generalized to reproducing kernel Hilbert spaces as

$$\begin{aligned} \min_{\boldsymbol{W},\boldsymbol{S}} f(\boldsymbol{W},\boldsymbol{S}) &= \|\phi(\boldsymbol{X}) - \phi(\boldsymbol{X})\boldsymbol{W}\boldsymbol{S}\|_H^2 + 2\alpha\|\boldsymbol{S}\|_1 \\ s.t.\ \ &\|\phi(\boldsymbol{X})\boldsymbol{W}_{\bullet k}\|_H \leq 1,\ \forall k = 1, 2, \ldots, K, \end{aligned} \tag{3}$$

which is single-view self-explanatory sparse representation (SSSR).

Now, the Frobenius norm has been replaced by the inner-product norm of that Hilbert space, such that $||\phi(\boldsymbol{X})||_H^2 = \kappa(\boldsymbol{X}, \boldsymbol{X})$, with kernel function $\kappa(\boldsymbol{X}_{\bullet i}, \boldsymbol{X}_{\bullet j}) = \phi(\boldsymbol{X}_{\bullet i})^T \phi(\boldsymbol{X}_{\bullet j})$. The dictionary becomes a set of K arbitrary functions in that Hilbert space. Using the "*kernel trick*", we get

$$\begin{aligned} &\|\phi(\boldsymbol{X}) - \phi(\boldsymbol{X})\boldsymbol{W}\boldsymbol{S}\|_H^2 + 2\alpha\|\boldsymbol{S}\|_1 \\ &= trace\{\kappa(\boldsymbol{X},\boldsymbol{X})\} - 2trace\{\kappa(\boldsymbol{X},\boldsymbol{X})\boldsymbol{W}\boldsymbol{S}\} \\ &\quad + trace\{\boldsymbol{S}^T\boldsymbol{W}^T\kappa(\boldsymbol{X},\boldsymbol{X})\boldsymbol{W}\boldsymbol{S}\} + 2\alpha\|\boldsymbol{S}\|_1. \end{aligned} \tag{4}$$

On the other hand, if directly kernelizing the primal form (1), we get

$$\min_{\boldsymbol{B},\boldsymbol{S}} f(\boldsymbol{B},\boldsymbol{S}) = \|\phi(\boldsymbol{X}) - \phi(\boldsymbol{B})\boldsymbol{S}\|_H^2 + 2\alpha\|\boldsymbol{S}\|_1. \tag{5}$$

Still, according to the representer theorem [26], the solution $\phi(\boldsymbol{B})$ to problem (5) has the form $\phi(\boldsymbol{B}) = \phi(\boldsymbol{X})\boldsymbol{W}$. This is already explicitly encoded in the formulation (3). That is, Eqns. 5, 3 are intuitively akin to the primal and dual forms of sparse representation in the Hilbert spaces.

There are also some benefits which make the dual form (3) preferable. Exactly optimizing to the standard formulation (5) is quite difficult. In the new high dimensional space, t, the dimension of $\phi(\boldsymbol{X}) \gg$ the number of samples N, perhaps even infinite. By leveraging the "kernel trick", this can only be partially tackled.

Since $\boldsymbol{B}$ is involved in $\kappa(\boldsymbol{B}, \boldsymbol{B})$, the optimal solution to $\boldsymbol{B}$ is always related to the partial derivative of $\kappa(\boldsymbol{B}, \boldsymbol{B})$ with respect to $\boldsymbol{B}$, which is relatively easy only for some specific kernels [4]. For others, only an approximation strategy is feasible, where the dictionary in the kernel space is transformed from the one learned in the original space [5]. There is no guarantee that the transformation of the optimal dictionary in the original space will remain optimal in the kernel space. However, using the equivalent formulation (3), we can now search an optimal dictionary directly in the kernel space through optimizing $\boldsymbol{W}$ instead of $\boldsymbol{B}$. Since (4) only depends on the kernel function $\kappa(\boldsymbol{X},\boldsymbol{X}) = \phi(\boldsymbol{X})^T\phi(\boldsymbol{X})$, which can be precomputed before sparse representation, we can now handle arbitrary kernels with tractable computation.

3.2 Multiple View Joint Formulation

Using a single specific kernel may be a source of bias, and in allowing a learner to combine a set of kernels, a better solution can be found. Here, instead of choosing a single kernel function, a feasible alternative is to use a combination of kernels as in multiple kernel learning (MKL) methods.

Assume there are G candidate Hilbert spaces forming a set as $\mathcal{H} = \{H_1, ..., H_g, ..., H_G\}$, and the corresponding kernel functions $\left\{\kappa_g : \mathbb{R}^{D_g} \times \mathbb{R}^{D_g} \rightarrow \mathbb{R}\right\}_{g=1}^{G}$ with $\kappa_g(\boldsymbol{X}_{\bullet i}, \boldsymbol{X}_{\bullet j}) = \phi_g(\boldsymbol{X}_{\bullet i})^T\phi_g(\boldsymbol{X}_{\bullet j})$. Candidate spaces include the well-known linear kernel space, the polynomial kernel space, the Gaussian RBF kernel, and widely used ones in vision community such as the Hellinger kernel space and the histogram intersection kernel space. Given the original G feature representations $\boldsymbol{X}^g$ with dimension $D_g \times N$ (not necessarily different) of data instances and mapping them to these different Hilbert spaces, the general formulation for multiple kernel learning sparse representation is

$$\kappa_\eta\left(\boldsymbol{X}_{\bullet i}, \boldsymbol{X}_{\bullet j}\right) = f_\eta\left(\left\{\kappa_g\left(\boldsymbol{X}^g_{\bullet i}, \boldsymbol{X}^g_{\bullet j}\right)_{g=1}^{G}\right\}\right), \tag{6}$$

where $f_\eta : \mathbb{R}^G \rightarrow \mathbb{R}$ is a linear or nonlinear function combination function. The weight matrix $\boldsymbol{W}$ and $\boldsymbol{S}$ are also redefined in different spaces as $\boldsymbol{W} = \{\boldsymbol{W}^g\}_{g=1}^{G}$ and $\boldsymbol{S} = \{\boldsymbol{S}^g\}_{g=1}^{G}$.

For visual tasks, $\boldsymbol{S}$ is the most important part in that it serves as the newly mapped feature representation and the input of the final classifiers. Since different kernels correspond to different notions of similarity, $\{\boldsymbol{S}^g\}_{g=1}^{G}$ in different Hilbert spaces will capture various salient regions or structures, making the final representation more discriminative. Here, we fix the input features from a single source, and focus on its combination, and then generalize (3) to multiple-view self-explanatory sparse representation (MSSR):

$$\begin{aligned} \min_{\boldsymbol{W},\boldsymbol{S}} f(\boldsymbol{W}, \boldsymbol{S}) = & \sum_{H_g \in \mathcal{H}} \|\phi^g(\boldsymbol{X}) - \phi^g(\boldsymbol{X})\boldsymbol{W}^{\boldsymbol{g}}\boldsymbol{S}^{\boldsymbol{g}}\|^2_{H_g} + 2\alpha\|\boldsymbol{S}^g\|_1 \\ & s.t. \|\phi^g(\boldsymbol{X})\boldsymbol{W}^g{}_{\bullet k}\|_{H_g} \leq 1,\ \forall k = 1, \ldots, K, g = 1, \ldots, G. \end{aligned} \tag{7}$$

After obtaining $\{\boldsymbol{S}^g\}$, we concatenate them to form the final representation as $\boldsymbol{S}$. Another notable benefit is that since each set $\{\boldsymbol{W}^g, \boldsymbol{S}^g\}$ can be learned and inferred independently from each other, the computational cost is significantly reduced if a large weight matrix $\boldsymbol{W}$ is required. Generally speaking, for sparse representation a larger dictionary will lead to better performance while the computational consumption grows beyond linear increase. Moreover, since our dictionary blocks are built from different kernel spaces, it will outperform the one coming from the same kernel space. In our experiments, we show that learning 4,096 bases in total by learning four sets of 1,024 bases separately, outperforms 4,096 bases obtained from the single kernel space both in speed and classification accuracy.

4 Optimization of the Objective Function

In this section, we focus on solving the optimization of the objective function proposed in the last section. This optimization problem is not jointly convex in both $\boldsymbol{W}^g$ and $\boldsymbol{S}^g$, but is separately convex in either $\boldsymbol{W}^g$ or $\boldsymbol{S}^g$ with $\boldsymbol{S}^g$ or $\boldsymbol{W}^g$ fixed. So the objective function can be optimized by alternating minimization to two optimization subproblems as follows.

– With fixed $\boldsymbol{W}^g$, the objective function of finding sparse codes $\boldsymbol{S}^g$ can be written as an ℓ_1- regularized least-squares $(\ell_1 - ls)$ minimization subproblem:

$$f(\boldsymbol{S}^g) = \|\phi^g(\boldsymbol{X}) - \phi^g(\boldsymbol{X})\boldsymbol{W}^g\boldsymbol{S}^g\|_F^2 + 2\alpha\|\boldsymbol{S}^g\|_1 \tag{8}$$

– With fixed $\boldsymbol{S}^g$, the objective function of learning weight $\boldsymbol{W}^g$ can be written as an ℓ_2- constrained least-squares $(\ell_2 - ls)$ minimization subproblem:

$$\begin{aligned} f(\boldsymbol{W}^g) &= \|\phi^g(\boldsymbol{X}) - \phi^g(\boldsymbol{X})\boldsymbol{W}^g\boldsymbol{S}^g\|_F^2 \\ s.t.&\ \|\phi^g(\boldsymbol{X})\boldsymbol{W}^g{}_{\bullet k}\|_2^2 \le 1,\ \forall k = 1, 2, \cdots, K. \end{aligned} \tag{9}$$

4.1 $\ell_1 - ls$ Minimization Subproblem

Eqn. 8 can be simplified as

$$\begin{aligned} f(\boldsymbol{S}^g) = trace\{\kappa^g(\boldsymbol{X}, \boldsymbol{X})\} - 2\sum_{n=1}^{N}[\kappa^g(\boldsymbol{X}, \boldsymbol{X})\boldsymbol{W}^g]_{n\bullet}\boldsymbol{S}^g{}_{\bullet n} \\ + \sum_{n=1}^{N}\boldsymbol{S}^{gT}_{\bullet n}[\boldsymbol{W}^{gT}\kappa^g(\boldsymbol{X}, \boldsymbol{X})\boldsymbol{W}^g]\boldsymbol{S}^g{}_{\bullet n} + 2\alpha\sum_{k=1}^{K}\sum_{n=1}^{N}|\boldsymbol{S}^g{}_{kn}|. \end{aligned} \tag{10}$$

For each feature $\boldsymbol{x}$ in $\boldsymbol{X}$, the objective function in Eqn. 10 can be rewritten as

$$f(\boldsymbol{s}^g) = \kappa^g(\boldsymbol{x}, \boldsymbol{x}) + \boldsymbol{s}^{gT}\boldsymbol{U}\boldsymbol{s}^g - 2\boldsymbol{V}\boldsymbol{s}^g + 2\alpha\|\boldsymbol{s}^g\|_1, \tag{11}$$

where $\boldsymbol{U} = \boldsymbol{W}^{gT}\kappa^g(\boldsymbol{X}, \boldsymbol{X})\boldsymbol{W}^g$, $\boldsymbol{V} = \kappa^g(\boldsymbol{x}, \boldsymbol{X})\boldsymbol{W}^g$. Once the $\boldsymbol{W}^g$ and $\kappa^g(\boldsymbol{X}, \boldsymbol{X})$ are fixed, we can easily extend the feature-sign search algorithm [13] to optimize the objective function.

Denoting $L(\boldsymbol{s}^g) = \kappa^g(\boldsymbol{x},\boldsymbol{x}) + \boldsymbol{s}^{gT}\boldsymbol{U}\boldsymbol{s}^g - 2\boldsymbol{V}\boldsymbol{s}^g$, then

$$\frac{\partial L(\boldsymbol{s}^g)}{\partial \boldsymbol{s}^g} = 2\boldsymbol{U}\boldsymbol{s}^g - 2\boldsymbol{V}^T, \tag{12}$$

$$\frac{\partial^2 L(\boldsymbol{s}^g)}{\partial^2 \boldsymbol{s}^g} = 2\boldsymbol{U}. \tag{13}$$

The sparse coding algorithm can be represented as solving the problem: $\min_{\boldsymbol{s}^g} L(\boldsymbol{s}^g)$ $+2\alpha\|\boldsymbol{s}^g\|_1$. The detailed algorithmic procedure uses Algorithm 1 in [5]. Note that the computational cost of SSSR or MSSR is the same as that of sparse coding in [13] except for the additional expenditure in calculating the different kernel matrix.

4.2 $\ell_2 - ls$ Minimization Subproblem

Ignoring the unrelated term, Eqn. 9 can be simplified as

$$\begin{aligned} f(\boldsymbol{W}^g) = & -2\sum_{k=1}^{K}[\boldsymbol{S}^g\kappa^g(\boldsymbol{X},\boldsymbol{X})]_{k\bullet}\boldsymbol{W}^g{}_{\bullet k} + \sum_{k=1}^{K} W^{gT}_{\bullet k}[\kappa^g(\boldsymbol{X},\boldsymbol{X})\boldsymbol{W}^{\boldsymbol{g}}\boldsymbol{S}^{\boldsymbol{g}}\boldsymbol{S}^{gT}]_{\bullet k} \\ & s.t.\ \|\phi^g(\boldsymbol{X})\boldsymbol{W}^g{}_{\bullet k}\|_2^2 \le 1, \forall k = 1,2,\cdots,K. \end{aligned} \tag{14}$$

We optimize each column of $\boldsymbol{W}^g$ alternately. Specifically, ignoring the constant term $trace\{\kappa^g(\boldsymbol{X},\boldsymbol{X})\}$, the Lagrangian is

$$\begin{aligned} \mathcal{L}(\boldsymbol{W}^g,\lambda_k) = & \sum_{k=1}^{K} W^{gT}_{\bullet k}[\kappa^g(\boldsymbol{X},\boldsymbol{X})\boldsymbol{W}^{\boldsymbol{g}}\boldsymbol{S}^{\boldsymbol{g}}\boldsymbol{S}^{gT}]_{\bullet k} - 2\sum_{k=1}^{K}[\boldsymbol{S}^g\kappa^g(\boldsymbol{X},\boldsymbol{X})]_{k\bullet}\boldsymbol{W}^g{}_{\bullet k} \\ & + \lambda_k(1 - [\boldsymbol{W}^{gT}\kappa^g(\boldsymbol{X},\boldsymbol{X})\boldsymbol{W}^g]_{kk}). \end{aligned} \tag{15}$$

The partial derivative with respect to $\boldsymbol{W}^g_{\bullet k}$ is

$$\frac{\partial \mathcal{L}(\boldsymbol{W}^g,\lambda_k)}{\partial \boldsymbol{W}^g{}_{\bullet k}} = \boldsymbol{0}. \tag{16}$$

Hence, the solution to $\boldsymbol{W}^g_{\bullet k}$ is obtained as

$$\boldsymbol{W}^g{}_{\bullet k} = \frac{\boldsymbol{S}^{gT}_{k\bullet} - [\widetilde{\boldsymbol{W}^g}^k\boldsymbol{F}]_{\bullet k}}{\boldsymbol{F}_{kk} - \lambda_k}, \tag{17}$$

where $\boldsymbol{F} = \boldsymbol{S}^g\boldsymbol{S}^{gT}$, $\widetilde{\boldsymbol{W}^g}^k = \begin{cases} \boldsymbol{W}^g{}_{\bullet p}, & p \ne k \\ \boldsymbol{0}, & p = k \end{cases}$. Now, substituting $\boldsymbol{W}^g_{\bullet k}$ into the Lagrangian and only keeping the term including $\boldsymbol{W}^g_{\bullet k}$, we then have

$$\mathcal{L}(\boldsymbol{W}^g,\lambda_k) = \lambda_k + \frac{(\boldsymbol{S}^T_{k\bullet} - [\widetilde{\boldsymbol{W}^g}^k\boldsymbol{F}]_{\bullet k})^T\kappa^g(\boldsymbol{X},\boldsymbol{X})(\boldsymbol{S}^{gT}_{k\bullet} - [\widetilde{\boldsymbol{W}^g}^k\boldsymbol{F}]_{\bullet k})}{\lambda_k - \boldsymbol{F}_{kk}}. \tag{18}$$

Thus, λ_k can be obtained. Substituting λ_k into $\boldsymbol{W}^g_{\bullet k}$,

$$\boldsymbol{W}^g{}_{\bullet k} = \frac{\boldsymbol{S}^{gT}_{k\bullet} - [\widetilde{\boldsymbol{W}^g}^k\boldsymbol{F}]_{\bullet k}}{\pm\sqrt{(\boldsymbol{S}^{gT}_{k\bullet} - [\widetilde{\boldsymbol{W}^g}^k\boldsymbol{F}]_{\bullet k})^T\kappa^g(\boldsymbol{X},\boldsymbol{X})(\boldsymbol{S}^{gT}_{k\bullet} - [\widetilde{\boldsymbol{W}^g}^k\boldsymbol{F}]_{\bullet k})}}. \tag{19}$$

From Eqn. 19, two solutions are obtained with $\pm$ signs. The sign of $\boldsymbol{W}^g_{\bullet k}$ is not essential since it can be easily absorbed by converting between $\boldsymbol{S}^g_{k\bullet}$ and $-\boldsymbol{S}^g_{k\bullet}$.

Algorithm 1 Algorithm for SSSR or MSSR

Require: Data matrix $\boldsymbol{X} \in \mathbb{R}^{D\times N}$, α and K
Compute the kernels $\kappa(\boldsymbol{X},\boldsymbol{X})$ on $\boldsymbol{X}$.
$\boldsymbol{W} \leftarrow rand(N,K) - 0.5$, $\boldsymbol{S} \leftarrow zeros(K,N) - 0.5$
Compute $\boldsymbol{F} = \boldsymbol{S}\boldsymbol{S}^T$,$\boldsymbol{G} = \boldsymbol{F} \odot (\boldsymbol{1} - \boldsymbol{I})$
for $k = 1; k \leq K; k{+}{+}$ **do**
  $\delta = \boldsymbol{W}_{\bullet k}^T \kappa(\boldsymbol{X},\boldsymbol{X}) \boldsymbol{W}_{\bullet k}$
  $\boldsymbol{W}_{\bullet k} = \boldsymbol{W}_{\bullet k} / \sqrt{\delta}$
end for
$iter = 0$
while $(f(iter) - f(iter+1))/f(iter) > 1e{-}5$ **do**
  $iter \leftarrow iter + 1$
  Update $\boldsymbol{S}$**:**
  Sparse coding: compute using feature-sign search algorithm
  Update $\boldsymbol{W}$**:**
  Compute $\boldsymbol{F} = \boldsymbol{S}\boldsymbol{S}^T$,$\boldsymbol{G} = \boldsymbol{F} \odot (\boldsymbol{1} - \boldsymbol{I})$
  for $k = 1; k \leq K; k{+}{+}$ **do**
    $\boldsymbol{W}_{\bullet k} = \boldsymbol{S}_{k\bullet}^T - \boldsymbol{W}\boldsymbol{G}_{\bullet k}$
    $\delta = \boldsymbol{W}_{\bullet k}^T \kappa(\boldsymbol{X},\boldsymbol{X}) \boldsymbol{W}_{\bullet k}$
    $\boldsymbol{W}_{\bullet k} = \boldsymbol{W}_{\bullet k} / \sqrt{\delta}$
  end for
  Update the objective function:
  $f = trace\{\kappa(\boldsymbol{X},\boldsymbol{X})\} - 2trace\{\boldsymbol{A}\boldsymbol{S}^T\} + trace\{\boldsymbol{F}\boldsymbol{E}\} + 2\alpha\|\boldsymbol{S}\|_1$
end while
return $\boldsymbol{W}$, and $\boldsymbol{S}$

4.3 Overall Algorithm

Our algorithm for SSSR or MSSR is shown in Algorithm 1. Here, $\boldsymbol{1} \in \mathbb{R}^{K\times K}$ is a square matrix with all elements 1, $\boldsymbol{I} \in \mathbb{R}^{K\times K}$ is the identity matrix, and $\odot$ indicates the Hadamard product. By iterating $\boldsymbol{S}$ and $\boldsymbol{W}$ alternately, the sparse codes are obtained, and the bases are learned.

5 Experimental Results

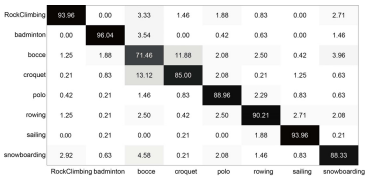
In this section, we present our experimental results for SSSR and MSSR compared with several baselines and previous published techniques on four benchmark datasets, such as UIUC-Sports dataset [15], Scene 15 dataset [12], Caltech-101 dataset [14], and Caltech-256 dataset [10].

5.1 Experimental Settings

For each dataset, the data are randomly split into training set and testing set based on published protocols. The experimental process is repeated 8 times, and the mean and standard deviation of the classification accuracy are record. Each image is resized with maximum side 300 pixels firstly, except 400 pixels for UIUC-Sports dataset due to the high resolution of original images. As for the image features, two types of densely sampled SIFT features are used to demonstrate the effectiveness of SSSR and MSSR. One feature is extracted with patch size 16×16 and step size 8 pixels, which we call single scale SIFT. The other one is extracted under three scales 16×16, 24×24, and 32×32, and the step size 8 pixels, which

Table 1. Performance comparisons on UIUC-Sports dataset and Scene 15 dataset (%).

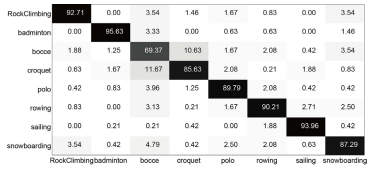
Methods	UIUC-Sports	Scene 15
Single scale SIFT		
ScSPM(1024) [40,6]	82.74 ± 1.46	80.28 ± 0.93
EMK [1]	74.56 ± 1.32	NA
KSR [4]	84.92 ± 0.78	83.68 ± 0.61
SCSR(1024) [17]	87.97 ± 1.11	81.51 ± 0.32
DLSM(1024) [18]	86.82 ± 1.04	83.40 ± 0.44
DLMM(1024) [18]	86.93 ± 0.99	83.67 ± 0.49
Ours(SSSR)		
Hellinger+1024+linearSVM	88.49 ± 1.25	82.25 ± 0.31
HIK+1024+linearSVM	88.41 ± 1.11	84.42 ± 0.33
POLY+1024+linearSVM	88.26 ± 1.12	83.59 ± 0.26
linear+1024+linearSVM	88.07 ± 1.33	83.84 ± 0.40
Ours(MSSR)		
4096+linearSVM	89.77 ± 1.12	85.18 ± 0.26
4096+polySVM	89.79 ± 0.96	**85.36 ± 0.29**
Multiple scale SIFT		
KSRSPM-HIK(4096)[5]	86.85 ± 0.45	NA
Ours(SSSR)		
Hellinger+4096+linearSVM	88.36 ± 0.82	84.89 ± 0.37
HIK+4096+linearSVM	88.54 ± 1.09	84.18 ± 0.47
POLY+4096+linearSVM	88.93 ± 0.81	84.09 ± 0.35
linear+4096+linearSVM	88.83 ± 0.81	83.67 ± 0.46
Ours(MSSR)		
16384+linearSVM	**89.95 ± 0.64**	84.89 ± 0.38
16384+polySVM	89.61 ± 0.70	84.93 ± 0.45



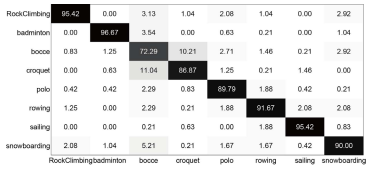
(a) Hellinger kernel

(b) HIK kernel

(c) polynomial kernel



(d) linear kernel



(e) MSSR

Fig. 2. Confusion matrix on UIUC-Sports dataset (%) with single scale SIFT features

Table 2. Performance comparisons on Caltech-101 dataset (%)

Methods	5train	10train	15train	20train	25train	30train
Single scale SIFT						
ScSPM(1024) [40]	NA	NA	67.0 ± 0.45	NA	NA	73.2 ± 0.54
DLSM(1024) [18]	NA	NA	66.88 ± 0.53	NA	NA	74.39 ± 0.82
DLMM(1024) [18]	NA	NA	67.54 ± 0.41	NA	NA	74.87 ± 0.67
Ours(SSSR)						
Hellinger+1024+linearSVM	47.42 ± 0.61	60.64 ± 0.48	65.65 ± 0.30	68.83 ± 0.50	71.35 ± 0.58	73.04 ± 1.27
HIK+1024+linearSVM	47.66 ± 0.41	60.44 ± 0.44	65.91 ± 0.54	69.05 ± 0.39	71.59 ± 0.73	73.43 ± 0.65
POLY+1024+linearSVM	48.10 ± 0.35	60.67 ± 0.37	65.91 ± 0.68	69.43 ± 0.21	71.77 ± 0.63	73.80 ± 0.64
linear+1024+linearSVM	48.27 ± 0.47	61.04 ± 0.59	66.26 ± 0.57	69.31 ± 0.65	71.72 ± 0.71	73.47 ± 0.42
Ours(MSSR)						
4096+linearSVM	49.52 ± 0.47	62.50 ± 0.23	67.97 ± 0.53	71.21 ± 0.38	73.68 ± 0.74	76.04 ± 0.67
4096+polySVM	49.34 ± 0.45	62.48 ± 0.26	67.79 ± 0.48	71.39 ± 0.36	73.63 ± 0.70	76.06 ± 0.83
Multiple scale SIFT						
LLC(4096) [34]	51.15	59.77	65.43	67.74	70.16	73.44
SC(AxMin@n)(4k) [11][1]	NA	NA	**74.6 ± 0.4**	NA	NA	**81.3 ± 0.6**
Ours(SSSR)						
Hellinger+4096+linearSVM	51.43 ± 0.82	64.60 ± 0.47	70.09 ± 0.27	73.70 ± 0.50	75.60 ± 0.51	77.43 ± 1.13
HIK+4096+linearSVM	51.81 ± 0.75	64.83 ± 0.56	69.93 ± 0.43	73.40 ± 0.57	75.25 ± 0.47	77.16 ± 1.01
POLY+4096+linearSVM	52.22 ± 1.02	65.39 ± 0.44	70.26 ± 0.50	73.79 ± 0.57	75.72 ± 0.48	77.31 ± 0.90
linear+4096+linearSVM	52.76 ± 0.81	65.67 ± 0.54	70.62 ± 0.60	74.18 ± 0.55	75.90 ± 0.60	77.51 ± 0.88
Ours(MSSR)						
16384+linearSVM	**53.36 ± 0.71**	**66.20 ± 0.56**	71.58 ± 0.43	**75.23 ± 0.68**	**76.89 ± 0.60**	78.74 ± 0.81
16384+polySVM	53.10 ± 0.76	66.10 ± 0.45	71.41 ± 0.38	75.08 ± 0.61	76.82 ± 0.52	78.59 ± 0.95

we call multiple scales SIFT. 128 dimensional SIFT descriptors are obtained and normalized to 1 with ℓ_2-norm. For learning the dictionaries, 30,000∼50,000 samples are used. For single scale SIFT, the dictionary size is 1,024 for each kernel space. For multiple scales SIFT, the dictionary size is 4,096. The spatial pyramid matching kernel is with 1, 4, and 16 segments. We use a max pooling strategy [40]. An image is represented by the concatenation of each segment and normalized to 1 with ℓ_2-norm.

We use four different kernels: the Hellinger kernel $\left(\kappa(\boldsymbol{x},\boldsymbol{y}) = \sum_{d=1}^{D}\sqrt{\mathbf{x}_d\mathbf{y}_d}\right)$, histogram intersection kernel $\left(\kappa(\boldsymbol{x},\boldsymbol{y}) = \sum_{d=1}^{D}\min\{\mathbf{x}_d,\mathbf{y}_d\}\right)$, polynomial kernel $\left(\kappa(\boldsymbol{x},\boldsymbol{y}) = (1+\boldsymbol{x}^T\boldsymbol{y})^p\right)$, and linear kernel $\left(\kappa(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{x}^T\boldsymbol{y}\right)$. Here, we set $p = 2$. Now only one parameter α needs tuning in the objective functions of SSSR and MSSR. The choice of α is obtained by cross-validation (CV). The CV results indicated that the optimal performance is achieved when maintaining approximate 10 non-0 elements, which agrees with the empirical conclusion in [40]. The parameter α is 0.15 for linear kernel, 0.3 for polynomial kernel, 0.4 for histogram intersection kernel, and 0.5 for hellinger kernel. Linear or Polynomial kernel SVM classifier is used with one-vs-all multi-class, and the LIBSVM [2] package is used.

[1] In [11], the image features are extracted with $16, 24, 32, 40$ patch size and $4, 6, 8, 10$ step size, respectively. Besides, the experimental setting in [11] is "approximate pooling (AxMin@n) with 4 levels of SPM".

Table 3. Performance comparisons on Caltech-256 dataset (%)

Methods	15train	30train	45train	60train
Single scale SIFT				
ScSPM(1024) [40]	27.73 ± 0.51	34.02 ± 0.35	37.46 ± 0.55	40.14 ± 0.91
LLC(1024) [34,6]	27.74 ± 0.32	32.07 ± 0.24	35.09 ± 0.44	37.79 ± 0.42
KSR(1024) [4]	29.77 ± 0.14	35.67 ± 0.10	38.61 ± 0.19	40.30 ± 0.22
SCSR(1024) [17]	29.23 ± 0.38	35.51 ± 0.32	38.68 ± 0.29	41.05 ± 0.42
DLSM(1024) [18]	29.31 ± 0.58	35.12 ± 0.34	37.62 ± 0.57	39.96 ± 0.62
DLMM(1024) [18]	30.35 ± 0.42	36.22 ± 0.33	38.97 ± 0.56	41.09 ± 0.44
Ours(SSSR)				
Hellinger+1024+linearSVM	32.74 ± 0.35	39.68 ± 0.33	43.18 ± 0.41	45.33 ± 0.34
HIK+1024+linearSVM	32.38 ± 0.47	39.13 ± 0.48	42.40 ± 0.34	44.86 ± 0.32
POLY+1024+linearSVM	31.58 ± 0.22	38.32 ± 0.32	41.74 ± 0.47	44.24 ± 0.43
linear+1024+linearSVM	31.52 ± 0.31	38.19 ± 0.33	41.39 ± 0.49	43.95 ± 0.63
Ours(MSSR)				
4096+linearSVM	34.06 ± 0.36	41.14 ± 0.43	44.72 ± 0.42	47.26 ± 0.43
4096+polySVM	35.38 ± 0.31	42.92 ± 0.46	46.88 ± 0.52	49.70 ± 0.43
Multiple scale SIFT				
LLC(4096) [34]	34.36	41.19	45.31	47.68
KSRSPM-HIK(4096)[5]	33.61 ± 0.34	40.63 ± 0.22	44.41 ± 0.12	47.03 ± 0.35
IFK [24][2]	34.7 ± 0.2	40.8 ± 0.1	45.0 ± 0.2	47.9 ± 0.4
Ours(SSSR)				
Hellinger+4096+linearSVM	37.11 ± 0.50	44.73 ± 0.37	48.65 ± 0.43	51.24 ± 0.60
HIK+4096+linearSVM	35.95 ± 0.36	43.45 ± 0.29	47.27 ± 0.33	49.96 ± 0.56
POLY+4096+linearSVM	35.54 ± 0.33	42.94 ± 0.40	46.70 ± 0.41	49.42 ± 0.62
linear+4096+linearSVM	35.66 ± 0.43	43.10 ± 0.28	46.98 ± 0.38	49.52 ± 0.60
Ours(MSSR)				
16384+linearSVM	37.12 ± 0.41	44.95 ± 0.38	48.89 ± 0.37	51.47 ± 0.72
16384+polySVM	**37.76 ± 0.25**	**45.70 ± 0.47**	**49.83 ± 0.18**	**52.81 ± 0.53**

5.2 UIUC-Sports Dataset

For the UIUC-Sports dataset [15], there are 8 classes with 1,579 images in total. We follow the common setup: 70 images per class are randomly selected as the training data, and 60 images per class for testing. Figure 2 shows the confusion matrices with single scale SIFT features. Table 1 shows the performance of different methods. Our proposed MSSR algorithm outperforms the traditional sparse representation based image classification [40] by 7.05% with single scale SIFT features.

5.3 Scene 15 Dataset

For the Scene 15 dataset [12], there are 15 classes with 4,485 images in total. We use an identical experimental setup as [12]: 100 images per class are randomly selected as the training data, and the rest for testing. Table 1 lists the comparisons of our SSSR and MSSR methods with previous work. Our proposed MSSR algorithm outperforms the traditional sparse representation based image classification [40] by 5.08% with single scale SIFT features.

[2] In [24], 5 scales are used for extracting the image features, and the total length of the vector to represent each image is $30k$.

5.4 Caltech-101 Dataset

The Caltech-101 dataset [14] contains 102 classes, one of which is the background. After removing the background class, the remaining 101 classes with 8,677 images in total are used for classification, with each class varying from 31 to 800 images. We follow the standard experiment setup for this dataset: 5, 10, 15, 20, 25, and 30 images per category are selected as the training set, and the rest for testing (the maximum is 50 images per category for testing). Table 2 shows performances of different methods. The best results reported in [11] are 74.6% and 81.3% with 15 and 30 images per class as the training set. With single scale SIFT features, our proposed MSSR algorithm outperforms the traditional sparse representation based image classification [40] by 0.97% and 2.86% for 15 and 30 training images per class, respectively.

5.5 Caltech-256 Dataset

The Caltech-256 dataset [10] contains 257 classes, one of which is the background. After removing the background class, the remaining 256 classes with a total of 29,780 images are used for classification. We follow the standard experimental setup for this dataset: 15, 30, 45, and 60 training images per category, and 25 testing images per category. Table 3 shows the performance of different methods. With single scale SIFT features, our proposed MSSR algorithm outperforms the traditional sparse representation based image classification [40] by 7.65%, 8.9%, 9.42% and 9.56% for 15, 30, 45 and 60 training images per class, respectively.

6 Conclusions

In this paper, motivated by the fact that sparse representation, kernel representation, and multiple kernel learning are powerful tools in discovering hidden structure of complex data, we proposed novel single- and multiple-view self-explanatory sparse representation (SSSR and MSSR) schemes. By leveraging a self-explanatory reformulation of sparse representation, where the bases lie in the span of the image features, the new formula is readily generalized into reproducing kernel Hilbert spaces for arbitrary kernels with computational tractability and conceptual interpretability. SSSR is capable of identifying both nonlinear structural information and sparse active components. The multiple-view joint representation not only captures various structure information of the image features under different kernels, but also reduces the complexity of dictionary learning. This leads to enhanced visual representation power as has been demonstrated by extensive experiments on image classification tasks.

Acknowledgment. This work was supported by the National Natural Science Foundation of P.R. China (No. 61271407, No. 61301242, No. 61171118), Shandong Provincial Natural Science Foundation, China (No. ZR2011FQ016), and the Fundamental Research Funds for the Central Universities (No. R1405012A).

References

1. Bo, L., Sminchisescu, C.: Efficient match kernel between sets of features for visual recognition. In: Proceedings of Advances in Neural Information Processing Systems, vol. 2, pp. 135–143. The MIT Press (2009)
2. Chang, C.C., Lin, C.J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology 2(3), 27:1–27:27 (2011)
3. Elhamifar, E., Vidal, R.: Sparse subspace clustering: Algorithm, theory, and applications. IEEE Transactions on Pattern Analysis and Machine Intelligence 35(11), 2765–2781 (2013)
4. Gao, S., Tsang, I.W.-H., Chia, L.-T.: Kernel sparse representation for image classification and face recognition. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part IV. LNCS, vol. 6314, pp. 1–14. Springer, Heidelberg (2010)
5. Gao, S., Tsang, I.W.H., Chia, L.T.: Sparse representation with kernels. IEEE Transactions on Image Processing 22(2), 423–434 (2013)
6. Gao, S., Tsang, I.H., Chia, L.T.: Laplacian sparse coding, hypergraph laplacian sparse coding, and applications. IEEE Transactions on Pattern Analysis and Machine Intelligence 35(1), 92–104 (2013)
7. van Gemert, J.C., Veenman, C.J., Smeulders, A.W., Geusebroek, J.M.: Visual word ambiguity. IEEE Transactions on Pattern Analysis and Machine Intelligence 32(7), 1271–1283 (2010)
8. van Gemert, J.C., Veenman, C.J., Smeulders, A.W., Geusebroek, J.M.: Visual word ambiguity. IEEE Transactions on Pattern Analysis and Machine Intelligence 32(7), 1271–1283 (2010)
9. Gönen, M., Alpaydın, E.: Multiple kernel learning algorithms. Journal of Machine Learning Research 12, 2211–2268 (2011)
10. Griffin, G., Holub, A., Perona, P.: Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology (2007)
11. Koniusz, P., Yan, F., Mikolajczyk, K.: Comparison of mid-level feature coding approaches and pooling strategies in visual concept detection. Computer Vision and Image Understanding 117(5), 479–492 (2013)
12. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: Proceedings of the 19th CVPR, vol. 2, pp. 2169–2178. IEEE (2006)
13. Lee, H., Battle, A., Raina, R., Ng, A.: Efficient sparse coding algorithms. In: Proceedings of Advances in Neural Information Processing Systems, pp. 801–808. MIT Press (2006)
14. Li, F.F., Fergus, R., Perona, P.: Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In: Workshop of the 17th CVPR, vol. 12, p. 178. IEEE (2004)
15. Li, L.J., Li, F.F.: What, where and who? classifying events by scene and object recognition. In: Proceedings of the 11th ICCV, pp. 1–8. IEEE (2007)
16. Liu, B.D., Wang, Y.X., Bin, S., Zhang, Y.J., Wang, Y.J.: Blockwise coordinate descent schemes for sparse representation. In: Proceedings of the 39th ICASSP, pp. 5304–5308. IEEE (2014)
17. Liu, B.D., Wang, Y.X., Shen, B., Zhang, Y.J., Wang, Y.J., Liu, W.F.: Self-explanatory convex sparse representation for image classification. In: Proceedings of Systems, Man, and Cybernetics (SMC). pp. 2120–2125. IEEE (2013)
18. Liu, B.D., Wang, Y.X., Zhang, Y.J., Shen, B.: Learning dictionary on manifolds for image classification. Pattern Recognition 46(7), 1879–1890 (2013)

19. Liu, B.D., Wang, Y.X., Zhang, Y.J., Zheng, Y.: Discriminant sparse coding for image classification. In: Proceedings of the 37th ICASSP, pp. 2193–2196. IEEE (2012)
20. Liu, G., Lin, Z., Yan, S., Sun, J., Yu, Y., Ma, Y.: Robust recovery of subspace structures by low-rank representation. IEEE Transactions on Pattern Analysis and Machine Intelligence 35(1), 171–184 (2013)
21. Liu, W., Tao, D.: Multiview hessian regularization for image annotation. IEEE Transactions on Image Processing 22(7), 2676–2687 (2013)
22. Liu, W., Tao, D., Cheng, J., Tang, Y.: Multiview hessian discriminative sparse coding for image annotation. Computer Vision and Image Understanding 118, 50–60 (2014)
23. Nguyen, H.V., Patel, V.M., Nasrabadi, N.M., Chellappa, R.: Kernel dictionary learning. In: Proceedings of the 37th ICASSP, pp. 2021–2024. IEEE (2012)
24. Perronnin, F., Sánchez, J., Mensink, T.: Improving the fisher kernel for large-scale image classification. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part IV. LNCS, vol. 6314, pp. 143–156. Springer, Heidelberg (2010)
25. Schölkopf, B., Smola, A., Müller, K.: Kernel principal component analysis. In: Gerstner, W., Hasler, M., Germond, A., Nicoud, J.-D. (eds.) ICANN 1997. LNCS, vol. 1327, pp. 583–588. Springer, Heidelberg (1997)
26. Schölkopf, B., Herbrich, R., Smola, A.J.: A generalized representer theorem. In: Helmbold, D.P., Williamson, B. (eds.) COLT 2001 and EuroCOLT 2001. LNCS (LNAI), vol. 2111, pp. 416–426. Springer, Heidelberg (2001)
27. Shen, B., Hu, W., Zhang, Y., Zhang, Y.J.: Image inpainting via sparse representation. In: Proceedings of the 34th ICASSP, pp. 697–700. IEEE (2009)
28. Shen, B., Si, L.: Non-negative matrix factorization clustering on multiple manifolds. In: Proceedings of the 24th AAAI, pp. 575–580. IEEE (2010)
29. Shen, B., Si, L., Ji, R., Liu, B.: Robust nonnegative matrix factorization via l_1 norm regularization. arXiv preprint arXiv:1204.2311 (2012)
30. Tan, H., Cheng, B., Feng, J., Feng, G., Wang, W., Zhang, Y.J.: Low-n-rank tensor recovery based on multi-linear augmented lagrange multiplier method. Neurocomputing 119, 144–152 (2013)
31. Tan, H., Cheng, B., Wang, W., Zhang, Y.J., Ran, B.: Tensor completion via a multi-linear low-n-rank factorization model. Neurocomputing 133, 161–169 (2014)
32. Thiagarajan, J., Ramamurthy, K., Spanias, A.: Multiple kernel sparse representations for supervised and unsupervised learning. IEEE Transactions on Image Processing 23(7), 2905–2915 (2014)
33. Vedaldi, A., Zisserman, A.: Sparse kernel approximations for efficient classification and detection. In: Proceedings of the 25th CVPR, pp. 2320–2327. IEEE (2012)
34. Wang, J., Yang, J., Yu, K., Lv, F., Huang, T., Gong, Y.: Locality-constrained linear coding for image classification. In: Proceedings of the 23rd CVPR, pp. 3360–3367. IEEE (2010)
35. Wang, Y.X., Gui, L.Y., Zhang, Y.J.: Neighborhood preserving non-negative tensor factorization for image representation. In: Proceedings of the 37th ICASSP, pp. 3389–3392. IEEE (2012)
36. Wang, Y.X., Zhang, Y.J.: Image inpainting via weighted sparse non-negative matrix factorization. In: Proceedings of the 18th ICIP, pp. 3409–3412. IEEE (2011)
37. Wang, Y.X., Zhang, Y.J.: Nonnegative matrix factorization: a comprehensive review. IEEE Transactions on Knowledge and Data Engineering 25(6), 1336–1353 (2013)

38. Wu, J., Rehg, J.M.: Beyond the euclidean distance: Creating effective visual codebooks using the histogram intersection kernel. In: Proceedings of the 12th ICCV, pp. 630–637. IEEE (2009)
39. Wu, Y., Shen, B., Ling, H.: Visual tracking via online non-negative matrix factorization. IEEE Transactions on Circuits and Systems for Video Technology 24(3), 374–383 (2014)
40. Yang, J., Yu, K., Gong, Y., Huang, T.: Linear spatial pyramid matching using sparse coding for image classification. In: Proceedings of the 22nd CVPR, pp. 1794–1801. IEEE (2009)
41. Yang, M., Zhang, L., Shiu, S.K., Zhang, D.: Robust kernel representation with statistical local features for face recognition. IEEE Transactions on Neural Networks and Learning Systems 24(6), 900–912 (2013)
42. Yuan, X.T., Yan, S.: Visual classification with multi-task joint sparse representation. In: Proceedings of the 23th CVPR, pp. 3493–3500. IEEE (2010)