Pattern Recognition Letters 78 (2016) 64–69

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Greedy dictionary learning for kernel sparse representation based classifier☆

Vinayak Abrol*, Pulkit Sharma, Anil Kumar Sao

SCEE, School of Computing and Electrical Engineering, Indian Institute of Technology, Mandi, India

ARTICLE INFO

Article history:
Received 27 August 2015
Available online 23 April 2016

Keywords:
Classification
Kernel sparse representations
Dictionary learning
Sparse coding

ABSTRACT

We present a novel dictionary learning (DL) approach for sparse representation based classification in kernel feature space. These sparse representations are obtained using dictionaries, which are learned using training exemplars that are mapped into a high-dimensional feature space using the kernel trick. However, the complexity of such approaches using kernel trick is a function of the number of training exemplars. Hence, the complexity increases for large datasets, since more training exemplars are required to get good performance for most of the pattern classification tasks. To address this, we propose a hierarchical DL approach which requires the kernel matrix to update the dictionary atoms only once. Further, in contrast to the existing methods, the dictionary is learned in a linearly transformed/coefficient space involving sparse matrices, rather than the kernel space. Compared to the existing state-of-the-art methods, the proposed method has much less computational complexity, but performs similar for various pattern classification tasks.

1. Introduction

In recent years kernel sparse representation based classifier (KSRC) has been widely explored in various pattern classification and recognition tasks [30,32,34]. These kernel algorithms were actually proposed to improve the performance of sparse representation based classifier (SRC) [27,33] which uses the linear modeling framework, by exploiting the advantages of projecting data in some high-dimensional space [15]. However, linear representations are inadequate for representing non-linear structures of the data which arise in many practical applications [32]. Hence, if an appropriate kernel function is utilized, then there is a high probability that similar features are grouped together in the high-dimensional space [34]. Linear modeling in the projected feature space can thus address the issue of non-linearity and provide better discrimination than their traditional counterparts in the original space. In addition, all KSRC based approaches can operate in a high-dimensional space without explicitly transforming the data in that space [15].

In KSRC, the aim is to obtain the sparse representation $\mathbf{a}_t$ of a test feature vector $\mathbf{x}_t$ which is mapped to some high-dimensional space, and classify it to the class that gives the smallest reconstruction error [15]. Given a dictionary $\ddot{\mathbf{D}}$ in higher dimensions (learned/build using class specific training data), the sparse vector is obtained by solving the following sparse coding problem [34]:

$$\underset{\mathbf{a}_t}{\operatorname{argmin}} \|\phi(\mathbf{x}_t) - \ddot{\mathbf{D}}\mathbf{a}_t\|_2^2 \text{ s.t } \forall_i \ \|\mathbf{a}_i\|_0 \leq T_0, \tag{1}$$

where $\|.\|_0$ is the l_0-norm (convex surrogates can also be used), T_0 denotes the imposed limit on the cardinality of the sparse vector. Here, the transformation function $\phi : \mathbb{R}^n \rightarrow \mathcal{S}$ maps the input space to a high-dimensional Hilbert space $\mathcal{S}$. However, in most cases the transformation ϕ is not known, and the optimization of problem in (1) is infeasible using traditional methods. This issue can be addressed by using a kernel similarity function κ, which avoids the explicit mapping of training data to space $\mathcal{S}$ [30]. Let $\mathcal{K}(\ddot{\mathbf{D}}, \ddot{\mathbf{D}})$ be a kernel matrix whose elements are computed using kernel κ as $\kappa(\mathbf{d}_i, \mathbf{d}_j) = \phi(\mathbf{d}_i)^T\phi(\mathbf{d}_j)$. Similarly let $\mathcal{K}(\ddot{\mathbf{D}}, \mathbf{x}_t)$ be a vector with elements $\kappa(\mathbf{d}_i, \mathbf{x}_t)$. Hence, (1) can be written as [34]:

$$\underset{\mathbf{a}_i}{\operatorname{argmin}} \kappa(\mathbf{x}_t, \mathbf{x}_t) + \mathbf{a}_t^T\mathcal{K}(\ddot{\mathbf{D}}, \ddot{\mathbf{D}})\mathbf{a}_t - 2\mathbf{a}_t^T\mathcal{K}(\ddot{\mathbf{D}}, \mathbf{x}_t) \quad \text{s.t } \|\mathbf{a}_t\|_0 \leq T_0. \tag{2}$$

The challenge with such a formulation is that the dictionary atom $\mathbf{d}_i$ in the original signal space corresponding to atom $\phi(\mathbf{d}_i)$ in space $\mathcal{S}$ is unknown [30]. To address this issue, the training matrix (or its subset) can be used as the dictionary [34]. However, using $\phi(\mathbf{X})$ as the dictionary $\ddot{\mathbf{D}}$ in KSRC is inefficient, especially when the number of training exemplars is very large [30,32]. For instance, (i) testing phase will be very slow, as determining sparse codes for

☆ This paper has been recommended for acceptance by Y. Liu

* Corresponding author. Tel.: +91 9646315212.
E-mail address: vinayak-abrol@students.iitmandi.ac.in (V. Abrol).

http://dx.doi.org/10.1016/j.patrec.2016.04.014
0167-8655/

Table 1
Matrix and vector dimensions.

$\mathbf{X}$	$\phi(\mathbf{X})$	$\ddot{\mathbf{D}}$	$\mathbf{B}$	$\mathbf{A}$	$\ddot{\mathbf{d}}_i$	$\mathbf{x}_i$	$\mathbf{a}_i$
$n \times l$	$\tilde{n} \times l$	$\tilde{n} \times m$	$l \times m$	$m \times l$	$\tilde{n} \times 1$	$n \times 1$	$m \times 1$

dictionaries with more number of atoms is computationally expensive, prohibiting real-time application, and (ii) manually selecting a subset of the training data to seed the dictionary is not only tedious but also sub-optimal since there is no guarantee that such selection form the best dictionary [18].

In order to address these issues, recent studies suggest in favor of learning a dictionary instead of using the training data itself [1,18,28,30,32]. The dictionary for each class is learned from its training signal set $\mathbf{X} \in \mathbb{R}^{n \times l}$ by minimizing the reconstruction error and satisfying the sparsity constraints [30]. Existing dictionary learning (DL) algorithms available in the literature solves the following optimization problem:

$$\underset{\ddot{\mathbf{D}},\mathbf{A}}{\operatorname{argmin}}\, g(\mathbf{A}) \text{ subject to } \|\phi(\mathbf{X}) - \ddot{\mathbf{D}}\mathbf{A}\|_F^2 \leq \epsilon, \tag{3}$$

where $\|.\|_F$ is the Frobenius norm, $g()$ is a function that promotes sparsity (e.g, l_0-norm), ϵ is the error tolerance constant, $\ddot{\mathbf{D}} \in \mathbb{R}^{\tilde{n} \times m}$ is a dictionary in space $\mathcal{S}$ and $\mathbf{A}$ is the sparse coefficient matrix corresponding to the transformed training set $\phi(\mathbf{X})$ [31]. This non-convex problem is solved via alternative minimization in two steps i.e., sparse coding and dictionary update. The sparse coding problem can be solved for each training similar to (2). Once the sparse code for each exemplar is calculated, the dictionary can be updated such that the error, $\|\phi(\mathbf{X}) - \ddot{\mathbf{D}}\mathbf{A}\|_F^2$ is minimized. DL gives a suitable number of discriminative atoms for each class spanning its signal space, but learning an optimal dictionary in space $\mathcal{S}$ is not straight forward as compared to signal space [30]. In [32], it has been proved that the optimal solution for the dictionary $\ddot{\mathbf{D}}$ has the form $\ddot{\mathbf{D}} = \mathbf{PB} = \phi(\mathbf{X})\mathbf{B}$. Here, $\mathbf{P} = \phi(\mathbf{X})$ is a better choice as compared to relying on any manual selection. Further, $\mathbf{P}$ acts as a known prior and a regularizer to reduce over-fitting and instability while DL. In fact with this formulation, the dictionary atoms are linear combinations of the training exemplars instead of the training data itself. Thus, one can tune the dictionary of a class via modifying its corresponding sparse matrix $\mathbf{B}$. To understand this, consider the objective function in (3) as:

$$\begin{aligned} &\|\phi(\mathbf{X}) - \phi(\mathbf{X})\mathbf{BA}\|_F^2 \\ &= \|\phi(\mathbf{X})(\mathbf{I} - \mathbf{BA})\|_F^2 \\ &= \mathbf{tr}((\mathbf{I} - \mathbf{BA})^T \mathcal{K}(\mathbf{X}, \mathbf{X})(\mathbf{I} - \mathbf{BA})) \end{aligned} \tag{4}$$

where $\mathcal{K}(\mathbf{X}, \mathbf{X})$ is the kernel matrix whose elements are computed as $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$. Such a formulation is more efficient to solve, since it involves a kernel matrix of finite dimensions and a sparse matrix $\mathbf{B}$, instead of dealing with a possibly very large or some high-dimensional dictionary $\ddot{\mathbf{D}}$. Here, some popular kernels such as linear, Gaussian and polynomial kernels can be employed [15]. For the reader's convenience, Table 1 summarizes the dimensions of all important matrices and vectors. Parameters involving $\tilde{n}$ are associated with the high-dimensional feature space.

In order to solve (4), both $\mathbf{B}$ and $\mathbf{A}$ are alternatively optimized with respect to the whole dictionary $\ddot{\mathbf{D}}$ using the kernel trick. Hence, existing algorithms have large time complexity due to: (i) size of the dataset, (ii) overcompleteness of the dictionary, and (iii) density and size of the kernel matrix. In some cases extracted features are such that the kernel matrix is sparse, and hence sparse matrix manipulation methods may be used. However, in most cases the kernel matrix is dense, which leads to increased time complexity mainly for large scale learning tasks. This is because kernel methods typically construct a kernel matrix $\mathcal{K} \in \mathbf{R}^{l \times l}$ where l is the number of training instances. Nevertheless, the complexity of dictionary optimization using the kernel trick is therefore a function of the number of training exemplars, instead of the dimensionality of the input exemplars. Further, optimizing $\mathbf{B}$ require the kernel matrix in each iteration of existing DL algorithms [32]. Although, to deal with large kernel matrices many methods have focused on computing its low-rank approximation [2,9], but the focus of this paper is to propose an effective approach only for alleviating memory and computational cost for DL.

In this work using the kernel trick, we show that alternative to (4), one can define an objective function such that learning $\mathbf{B}$ (separately for each class) is efficient and independent of any computations involving the kernel matrix. This is achieved by learning the matrix $\mathbf{B}$ in the coefficient space rather than the signal or kernel space. For this, the kernel DL problem is transformed into a more suitable form to find a numerically stable solution, a process referred to as *preconditioning* [25]. Note that, the sparse coding stage will still require the use of kernel matrix. In order to update the matrix $\mathbf{B}$, the proposed algorithm uses a hierarchical subset selection procedure. In each iteration, a column/atom of $\mathbf{B}$ is selected in accordance to its energy contribution, from the transformed training exemplars in the coefficient domain. It is done in such a way that the information learned by the previously updated atoms can be used to guide an adaptive design of subsequent atoms. Thus, after each update the modified residual serves as the new training set for the next update i.e., any atom is learned in accordance to what was not learned using previous atoms.

1.1. Related work

In earlier works of [15,20,34], the dictionaries used to compute the kernel sparse representation (based on l_1-norm minimization) consists of exemplars from the transformed training exemplars. In addition, $\mathcal{S}$ being a very high-dimensional space, in [20], the transformed exemplars are projected on to a reduced dimensionality subspace e.g., using principal component analysis (PCA). In contrast, works in [32] and [30], proposed to learn the dictionary by solving (4) using conventional DL approaches. In [30], the matrix $\mathbf{B}$ is updated using multilevel dictionary learning (MDL) method [29]. While in [32], $\mathbf{B}$ is updated based on Method of optimal directions (MOD) [12] or K-singular value decomposition (KSVD) [3], and the sparse coding stage is solved using the modified kernel OMP (KOMP) algorithm. In [32] $\mathbf{B}$ is optimized separately for different classes, while in [30] a single $\mathbf{B}$ is learned for all classes, using an ensemble of kernel matrices, such that the class discrimination is maximum. It is important to note that, the kernel MDL (KMDL), kernel MOD (KMOD) or kernel KSVD (KKSVD) formulation is highly non-convex and hard to solve in a moderate amount of time [32]. KMOD/KKSVD suffers from similar drawbacks as MOD/KSVD i.e., high complexity of the matrix inversion and lack of convergence guarantees, respectively. Moreover, in all the existing methods optimizing $\mathbf{B}$ require the kernel matrix in each iteration.

These issues are addressed in the proposed DL approach since: (i) it does not involve the kernel matrix to update the dictionary in each iteration of the algorithm, (ii) dictionary update is efficient as it is performed in the coefficient domain involving sparse matrices, and (iii) dictionary update does not involve any computationally intensive operations such as SVD or matrix inversion which makes it faster.

1.2. Organization of the paper

The rest of the paper is organized as follows: In Section 2 we propose an efficient algorithm for kernel sparse DL problem.