Confident Kernel Sparse Coding and Dictionary Learning

Babak Hosseini *
CITEC cluster of excellence
Bielefeld University, Germany
bhosseini@techfak.uni-bielefeld.de

Barbara Hammer
CITEC cluster of excellence
Bielefeld University, Germany
bhammer@techfak.uni-bielefeld.de

Abstract

In recent years, kernel-based sparse coding (K-SRC) has received particular attention due to its efficient representation of nonlinear data structures in the feature space. Nevertheless, the existing K-SRC methods suffer from the lack of consistency between their training and test optimization frameworks. In this work, we propose a novel confident K-SRC and dictionary learning algorithm (CKSC) which focuses on the discriminative reconstruction of the data based on its representation in the kernel space. CKSC focuses on reconstructing each data sample via weighted contributions which are confident in its corresponding class of data. We employ novel discriminative terms to apply this scheme to both training and test frameworks in our algorithm. This specif design increases the consistency of these optimization frameworks and improves the discriminative performance in the recall phase. In addition, CKSC directly employs the supervised information in its dictionary learning framework to enhance the discriminative structure of the dictionary. For empirical evaluations, we implement our CKSC algorithm on multivariate time-series benchmarks such as DynTex++ and UTKinect. Our claims regarding the superior performance of the proposed algorithm are justified throughout comparing its classification results to the state-of-the-art K-SRC algorithms.

Keywords: Discriminative dictionary learning, Kernel sparse coding, Non-negative reconstruction.

1 Introduction

SRC algorithms try to construct the input signals using weighted combinations of few selected entries from a set of learned prototypes. The vector of weighting coefficients and the set of prototypes are called the *sparse codes* and the *dictionary* respectively [2]. Thus, the resulting sparse representation can capture the essential intrinsic characteristics of the dataset [3]. Based on that property, discriminant SRC algorithms are proposed [4, 5] to learn a dictionary which can provide a discriminative representation of data classes. [6] showed that non-negative formulation of SRC could increase the possibility of relating each input signal to other resources from its class leading to a better classification accuracy.

Based on an implicit mapping of the data to a high-dimensional feature space, it is possible to formulate the kernel-based sparse coding (K-SRC) which notably enhances the reconstruction and discriminative abilities of the SRC framework [7, 8]. Nevertheless, the existing discriminant K-SRC models suffer from the lack of consistency between their training and the recall optimization models. The supervised information plays a crucial role in efficiently approximating the sparse

*Preprint of the publication [1], as provided by the authors. The final publication is available at IEEE Xplore via https://ieeexplore.ieee.org/document/8594939

codes in their training phase. Nevertheless, the lack of such information in the recall phase degrades the discriminative quality of the sparse coding model for test data.

1.1 Our Contributions

In this paper, we design a novel kernel-based sparse coding and dictionary learning for discriminative representation of the data. Our confident kernel sparse coding (CKSC) algorithm focuses on the discriminative reconstruction of each data point using other data inputs which are taken mostly from one class. Its kernel-SRC model is designed based upon a non-negative framework which facilitates such an intended representation. We emphasize the following contributions of our work which make it distinct from the relevant part of literature:

- We employ the supervised information in training and test part of our CKSC algorithm, and its recall phase has a consistent structure with its training model, which enhances its performance regarding the test data.
- The proposed non-negative discriminative sparse coding can learn each dictionary atom via constructing it upon mostly one class of data, but it can still be flexible enough to make small connections to other classes.
- Our discriminative dictionary learning method directly involves the supervised information in the optimization of both the dictionary and the sparse codes which make it more efficient regarding their discriminative objectives.

We review the relevant discriminative SRC methods in Section 2, and our proposed CKSC algorithm is introduced in Section 3. Its optimization steps are explained in Section 4, and the experiments and results are presented in Section 5. Lastly, the conclusion of the work is made in Section 6.

2 Background and Related Work

In this section, we briefly study relevant topics such as discriminant SRC and Kernel-based SRC.

2.1 Discriminative Sparse Coding

The training data matrix is denoted by $\mathbf{X} = [\vec{x}_1, ..., \vec{x}_N] \in \mathbb{R}^{d\times N}$, and we assume that the label information corresponding to $\mathbf{X}$ is given as $\mathbf{L} = [\vec{l}_1, \dots, \vec{l}_N] \in \mathbb{R}^{p\times N}$, where $\vec{l}_i$ is a binary vector such that $l_{iq} = 1$ if $\vec{x}_i \in \{class\ q\}$. Discriminative sparse coding is the idea of approximating each input signal as $\vec{x}_i \approx \mathbf{D}\vec{\gamma}_i$ while the cardinality norm $\|\mathbf{\Gamma}\|_0$ is as small as possible. The matrices $\mathbf{D} \in \mathbb{R}^{d\times c}$ and $\mathbf{\Gamma} = [\vec{\gamma}_1, \dots, \vec{\gamma}_N] \in \mathbb{R}^{c\times N}$ are the dictionary and sparse codes respectively. It is expected that the obtained $\mathbf{\Gamma}$ also respects the supervised information $\mathbf{L}$, such that each $\vec{l}_i$ can be predicted based on the corresponding $\vec{\gamma}_i$.

Typical discriminative sparse coding algorithms such as [4, 5, 9] focus on optimization problems generally similar to that of [4] as

$$\min_{\mathbf{\Gamma},\mathbf{D},\mathbf{W}} \quad \|\mathbf{X} - \mathbf{D\Gamma}\|_F^2 + \alpha f(\mathbf{L}, \mathbf{W}, \mathbf{U}, \mathbf{\Gamma}) + \lambda\|\mathbf{\Gamma}\|_1 \tag{1}$$

where $\|.\|_F$ refers to the Frobenius norm, and l_1-norm $\|\mathbf{\Gamma}\|_1$ is the relaxation of the l_0-norm which is advised by [10]. The objective $\|\mathbf{X} - \mathbf{D\Gamma}\|_F^2$ measures the quality of the reconstruction of $\mathbf{X}$ by dictionary $\mathbf{D}$. Objective function $f(.)$ reflects the classification error, and $\mathbf{W}$ denotes its matrix of parameters. The scalars α, λ specify the weights of the sparsity and discriminative terms respectively. In the framework of (1), the objective function $f(.)$ is considered to be jointly in terms of $(\mathbf{D}, \mathbf{\Gamma}, \mathbf{L})$. Nevertheless, in typical examples of discriminative SRC methods [4, 5], the update of $\mathbf{\Gamma}$ or $\mathbf{D}$ happens disjointed from the supervised information $\mathbf{L}$. Consequently, this often leads to low-quality local convergence points.

In some of the sparse coding works similar to [11, 12, 13, 14], the dictionary $\mathbf{D}$ is split into multiple class-specific sub-dictionaries which are separately trained to reconstruct each class of data. In [14] they learn a common dictionary module which is shared among all of the classes. Nevertheless, these methods usually face problems when different data classes are close to each other, and dictionary atoms from another class can also express some data points from one class.

Generally, a sparse coding framework which can incorporate the supervised training information in the recall phase provides a more efficient discriminant mapping for the test data. To our knowledge, the only discriminative sparse coding algorithm which merely aims for such consistency is the Fisher Discriminant sparse coding [13] which tries to reconstruct encodings of test data close to the average value of all $\vec{\gamma}_i$ related to the presumed class. However, in contrast to its base assumption, it is convenient for an SRC model to obtain distributed clusters of sparse codes, even though they are related to one class.

2.2 Kernel-based SRC

Incorporating kernel representation into sparse coding can extend it to nonlinear and non-vectorial domains [7, 15]. Accordingly, $\Phi : \mathbb{R}^d \rightarrow \mathbb{R}^h$ denotes as an implicit non-linear mapping which can transfer data to a reproducing kernel Hilbert space (RKHS). Therefore, we can use the kernel function $\mathcal{K}(\mathbf{X}, \mathbf{X})$ in the input space which is associated with the implicit mapping Φ such that $\mathcal{K}(\mathbf{X}, \mathbf{X}) = \langle \Phi(\mathbf{X}), \Phi(\mathbf{X}) \rangle$. Throughout using the kernel representation in the feature space, SRC can be reformulated similar to:

$$\min_{\mathbf{\Gamma}, \mathbf{D}} \|\Phi(\mathbf{X}) - \Phi(\mathbf{D})\mathbf{\Gamma}\|_F^2 \quad \text{s.t. } \|\vec{\gamma}_i\|_0 < T_0 \ \ \forall i \tag{2}$$

in which $\Phi(\mathbf{D})$ is the dictionary defined in the feature space.

It is shown by [7] that it is possible to define a dictionary in the feature space in the form of $\Phi(\mathbf{D}) = \Phi(\mathbf{X})\mathbf{U}$ where $\mathbf{U} \in \mathbb{R}^{N \times c}$. Each column of the dictionary matrix $\mathbf{U}$ contains a linear combination of data points from the feature space. However, to its advantage, the reconstruction term in (2) can be rephrased in terms of the Gram matrix $\mathcal{K}(\mathbf{X}, \mathbf{X})$ of the given data

$$\begin{aligned} &\|\Phi(\mathbf{X}) - \Phi(\mathbf{X})\mathbf{U}\mathbf{\Gamma}\|_F^2 = \\ &\mathcal{K}(\mathbf{X}, \mathbf{X}) + \mathbf{\Gamma}^\top \mathbf{U}^\top \mathcal{K}(\mathbf{X}, \mathbf{X})\mathbf{U}\mathbf{\Gamma} - 2\mathcal{K}(\mathbf{X}, \mathbf{X})\mathbf{U}\mathbf{\Gamma} \end{aligned} \tag{3}$$

which facilitates optimizing of dictionary matrix $\mathbf{U}$.

3 Confident Kernel Sparse Coding and Dictionary Learning

We propose a novel kernel-based discriminative sparse coding algorithm with the following training framework

$$\begin{aligned} Train: \quad \min_{\mathbf{\Gamma}, \mathbf{U}} \quad & \|\Phi(\mathbf{X}) - \Phi(\mathbf{X})\mathbf{U}\mathbf{\Gamma}\|_F^2 + \alpha \mathcal{F}(\mathbf{L}, \mathbf{\Gamma}, \mathbf{U}) \\ \text{s.t.} \quad & \|\vec{\gamma}_i\|_0 < T_0, \ \ \|\Phi(\mathbf{X})\vec{u}_i\|_2^2 = 1, \\ & \|\vec{u}_i\|_0 \leq T_0, \ \ u_{ij}, \gamma_{ij} \in \mathbb{R}^+, \ \ \forall ij \end{aligned} \tag{4}$$

and its relevant recall framework as

$$\begin{aligned} Recall: \quad \min_{\vec{\gamma}} \quad & \|\Phi(\vec{z}) - \Phi(\mathbf{X})\mathbf{U}\vec{\gamma}\|_F^2 + \alpha \mathcal{G}(\mathbf{L}, \vec{\gamma}, \mathbf{U}) \\ \text{s.t.} \quad & \|\vec{\gamma}\|_0 < T_0, \ \gamma_i \in \mathbb{R}^+ \ \ \forall i \end{aligned} \tag{5}$$

Similar to the work of [7], $\mathbf{\Gamma}$ and $\mathbf{U}$ denote sparse codes and the dictionary matrix of this framework respectively. The first part of the objectives in (9) and (10)) is the reconstruction loss function used in (3), and $\{\mathcal{F}, \mathcal{G}\}$ are the novel discriminative loss terms we introduce in this paper. Parameter T_0 applies the l_0-norm sparsity constraint on the columns of $\{\mathbf{U}, \mathbf{\Gamma}\}$, and α is the control factor between the reconstruction and the discriminant terms.

Furthermore, we can write $\Phi(\mathbf{X})\mathbf{U}\vec{\gamma} = \Phi(\mathbf{X})\vec{s}$ where $\vec{s} \in \mathbb{R}^N$. Therefore, by using the dictionary structure of $\Phi(\mathbf{X})\mathbf{U}$, it is also necessary to bound the value of $\|\vec{u}_i\|_0$. That way, each $\Phi(\vec{x})$ is encoded by selecting a few contributions from the training set $\mathbf{X}$ (small $\|\vec{s}\|_0$). Furthermore, the constraint $\|\Phi(\mathbf{X})\vec{u}_i\|_2^2 = 1$ is a bound on l_2-norm of the dictionary columns as a convenient way to prevent the solution of (4) from becoming degenerated [2].

In the following sub-sections, we discuss the mathematical detail of the novel objective terms employed in (9) and (10) and explain the motivations behind our such choice of design.

3.1 Discriminative Objective $\mathcal{F}(\mathbf{L}, \mathbf{U}, \mathbf{\Gamma})$:

Before discussing the mathematical content of $\mathcal{F}(\mathbf{L}, \mathbf{U}, \mathbf{\Gamma})$, we like to explain the motivation behind our specific choice of $\mathcal{F}$ as the discriminant term. If $\Phi(\vec{x})$ is reconstructed as $\Phi(\vec{x}) = \Phi(\mathbf{X})\mathbf{U}\vec{\gamma}$, then the entries of $\mathbf{L}\mathbf{U}\vec{\gamma} \in \mathcal{R}^c$ show the share of each class in the reconstruction of $\Phi(\vec{x})$. Accordingly, we denote each s-th row of the labeling matrix $\mathbf{L}$ as $\vec{\rho}_s^\top$ such that $\rho_{si} = 1$ if $\vec{x}_i$ belongs to class s. Now, as an extreme case, if we assume that $\vec{x}$ belongs to the class q and $\Phi(\vec{x})$ is lying on the subspace of class q in the feature space, we have $\vec{\rho}_s^\top \mathbf{U}\vec{\gamma} = 0 \; \forall s \neq q$. By generalizes this extreme case to a more realistic condition we define $\mathcal{F}(\mathbf{L}, \mathbf{U}, \vec{\gamma}) = \sum_{s \neq q} \rho_s^\top \mathbf{U}\vec{\gamma}$ as the sum of contributions from other classes (than q). Hence, for all $\mathbf{\Gamma}$ we have

$$\mathcal{F}(\mathbf{L}, \mathbf{U}, \mathbf{\Gamma}) = \mathrm{Tr}((\mathbf{1} - \mathbf{L}^\top)\mathbf{L}\mathbf{U}\mathbf{\Gamma}) \tag{6}$$

where $\mathbf{1} \in \mathbb{R}^{N \times c}$ is a matrix of all-ones, and $\mathrm{Tr}(.)$ denotes the matrix trace. Function $\mathcal{F}(\mathbf{L}, \mathbf{U}, \mathbf{\Gamma})$ is a linear term with respect to each $\vec{\gamma}_i$ and $\vec{u}_i$ individually. Therefore, it does not violate the convexity of the optimization problem in (4). Considering the optimization framework of (9), $\mathcal{F}$ is employed along with the additional term $\beta\|\mathbf{U}\mathbf{\Gamma}\|_F^2$. This term preserves the consistency between the training and the recall models and is explained in the next subsection.

3.2 Discriminative Recall Term $\mathcal{G}(\mathbf{L}, \vec{\gamma}, \mathbf{U})$:

For a test vector $\vec{z}$, we assume that $\Phi(\vec{z}) \in span\{\Phi(\mathbf{X})\}$ and belongs to the class q such that its projection on subspace q as $\|\Phi(\vec{z})^q\|_2$ is arbitrarily larger than $\|\Phi(\vec{z})\|_2 - \|\Phi(\vec{z})^q\|_2$. Therefore, via using the learned $\mathbf{U}$ from Eq (4), there exists a $\vec{\gamma}$ which reconstructs the test data as $\Phi(\vec{z}) = \Phi(\mathbf{X})\mathbf{U}\vec{\gamma}$ with more contributions chosen from the class q. Consequently, the class label $\vec{l}_z$ is predicted as $l_z(j) = 1$ where

$$j = \underset{j}{\mathrm{argmax}} \; \vec{\rho}_j^\top \mathbf{U}\vec{\gamma} \tag{7}$$

In other words, $\vec{l}_z$ is determined as the class of data which has the most contribution to the reconstruction of $\vec{z}$.

Since we do not have access to the labeling information for the test data, we propose a cross-entropy-like loss for (5) as

$$\mathcal{G}(\mathbf{L}, \vec{\gamma}, \mathbf{U}) = \sum_i (\sum_{s \neq i} h_s) h_i \quad \text{where } h_i := \rho_i^\top \mathbf{U}\vec{\gamma} \tag{8}$$

Since $\{\vec{\gamma}, \mathbf{U}\}$ are non-negative matrices, $\mathcal{G}$ is non-negative as well and can have its global optima where $\mathcal{G}(\vec{\gamma}^*) = 0$. Denoting $\vec{h}^* = \mathbf{L}\mathbf{U}\vec{\gamma}^*$, besides the trivial point of $\vec{h}^* = 0$, the loss term reaches its global optima when $\vec{h}^*$ contains only one non-zero value in its i-th entry. This observation is equivalent to finding $\vec{\gamma}^*$ such that it reconstructs $\vec{z}$ using contributions only from one class of data. Consequently, the non-trivial minima of both regularization terms in (6) and (8) occur at similar points where the decision vector $\mathbf{L}\mathbf{U}\vec{\gamma}$ has approximately a crisp form regarding only one of its entries. Therefore, adding $\mathcal{G}$ increases the consistency between training and the test frameworks.

By re-writing $\mathcal{G}(\mathbf{L}, \vec{\gamma}, \mathbf{U}) = \vec{\gamma}^\top \mathbf{U}^\top \mathbf{L}^\top (\mathbf{1} - \mathbf{I})\mathbf{L}\mathbf{U}\vec{\gamma}$, it is possible to show that $\mathcal{G}$ is a non-convex function due to the existing term $(\mathbf{1} - \mathbf{I})$ in its formulation. To fix this issue, we add $\beta\|\mathbf{U}\vec{\gamma}\|_2^2$ to the objective of (5) which converts its second-degree terms to

$$\vec{\gamma}^\top \mathbf{U}^\top (\mathbf{V} + \beta \mathbf{I}_{N \times N})\mathbf{U}\vec{\gamma}$$

where $\mathbf{V} := \mathcal{K}(\mathbf{X}, \mathbf{X}) + \alpha \mathbf{L}^\top (\mathbf{1} - \mathbf{I}_{c \times c})\mathbf{L}$. Now, denoting $\{\lambda_i\}_{i=1}^N$ as the eigenvalues of $\mathbf{V}$, we choose $\beta = -\min_i \lambda_i$ makes $(\mathbf{V} + \beta \mathbf{I}_{N \times N})$ a positive semi-definite matrix (PSD) and consequently, the whole objective becomes PSD due to its quadratic form. Therefore, (5) becomes a convex problem by adding this term.

In order to preserve the consistency between the test and training model, we also add the term $\beta\|\mathbf{U}\mathbf{\Gamma}\|_F^2$ to the discriminant loss $\mathcal{F}$ of (6) which results in (9). Doing so, we want to make sure the trained dictionary $\mathbf{U}$ has a proper structure also regarding (5). Furthermore, parameter β is independent of the test data and is computed only once before the start of the optimization phase. In the next section, we explain the optimization steps regarding frameworks of (4) and (5).

4 Optimization Scheme

By re-writing (4) and (5) using the provided descriptions of $\mathcal{F}$ and $\mathcal{G}$ in section 3, we obtain the following training optimization framework

$$\begin{aligned} Train: \quad \min_{\mathbf{\Gamma},\mathbf{U}} \quad & \|\Phi(\mathbf{X}) - \Phi(\mathbf{X})\mathbf{U}\mathbf{\Gamma}\|_F^2 + \beta\|\mathbf{U}\mathbf{\Gamma}\|_F^2 \\ & +\alpha \mathrm{Tr}\{(\mathbf{1} - \mathbf{L}^\top)\mathbf{L}\mathbf{U}\mathbf{\Gamma}\} \\ \text{s.t.} \quad & \|\vec{\gamma}_i\|_0 < T_0, \;\; \|\Phi(\mathbf{X})\vec{u}_i\|_2^2 = 1, \\ & \|\vec{u}_i\|_0 \leq T_0, \;\; u_{ij}, \gamma_{ij} \in \mathbb{R}^+, \;\; \forall ij \end{aligned} \tag{9}$$

and its relevant recall problem as

$$\begin{aligned} Test: \quad \min_{\vec{\gamma}} \quad & \|\Phi(\vec{z}) - \Phi(\mathbf{X})\mathbf{U}\mathbf{\Gamma}\|_F^2 + \beta\|\mathbf{U}\mathbf{\Gamma}\|_F^2 \\ & +\alpha(\vec{\gamma}^\top \mathbf{U}^\top \mathbf{L}^\top (\mathbf{1} - \mathbf{I})\mathbf{L}\mathbf{U}\vec{\gamma}) \\ \text{s.t.} \quad & \|\vec{\gamma}\|_0 < T_0, \; \gamma_i \in \mathbb{R}^+ \;\; \forall i \end{aligned} \tag{10}$$

Although the optimization problem (9) is not convex w.r.t $\{\mathbf{U}, \mathbf{\Gamma}\}$ together, We train the discriminant sparse coding model in 2 alternating convex optimization steps. At each step, we update one of the parameters while fixing the other one as presented in Algorithm 1.

4.1 Update of the Sparse Codes $\mathbf{\Gamma}$

The entire objective function in (9) has a separable column structure with respect to $\mathbf{\Gamma}$, and it can be optimized for each $\vec{\gamma}_i$ individually. Therefore, after removing the constant terms, (9) is re-written w.r.t each $\vec{\gamma}_i$ as

$$\begin{aligned} \min_{\vec{\gamma}_i} \quad & \vec{\gamma}_i^\top \big[\mathbf{U}^\top(\mathcal{K} + \beta\mathbf{I})\mathbf{U}\big]\vec{\gamma}_i \\ & +\big[\alpha(\mathbf{1} - \vec{l}_i^\top)\mathbf{L}\mathbf{U} - 2\mathcal{K}(\vec{x}_i, \mathbf{X})\mathbf{U}\big]\vec{\gamma}_i \\ \text{s.t.} \quad & \|\vec{\gamma}_i\|_0 < T_0, \;\; \gamma_{ij} \in \mathbb{R}^+, \;\; \forall ij \end{aligned} \tag{11}$$

where $\mathcal{K}$ stands for $\mathcal{K}(\mathbf{X}, \mathbf{X})$. This optimization framework is a non-negative quadratic programming problem with the constraint $\|\vec{\gamma}_i\| < T_0 \; \forall i$. Furthermore, $\mathcal{K} + \beta\mathbf{I}$ is a PSD matrix, and consequently (11) is a convex problem.

Therefore, we can employ the Non-Negative Quadratic Pursuit (NQP) algorithm [16] to optimize (11). NQP algorithm generalizes the Matching Pursuit approach [2] for quadratic problems similar to (11).

4.2 Update of the Dictionary $\mathbf{U}$

Similar to (11), it is also possible to reformulate the objective terms of (9) w.r.t. each dictionary column $\vec{u}_i$ separately. Accordingly, the reconstruction loss in (9) is equivalent to

$$\|\Phi(\mathbf{X})\mathbf{E}_i - \Phi(\mathbf{X})\vec{u}_i\vec{\gamma}^i\|_F^2, \;\; \mathbf{E}_i = (\mathbf{I} - \sum_{j \neq i} \vec{u}_j\vec{\gamma}^j) \tag{12}$$

in which $\vec{\gamma}^i$ is the i-th row of $\mathbf{\Gamma}$. Likewise, the rest of the objective term in (9) can be written in terms of $\vec{u}_i$ as

$$\beta\vec{u}_i^\top(\vec{\gamma}^i\vec{\gamma}^{i\top}\mathbf{I})\vec{u}_i + \alpha\vec{\gamma}^i(\mathbf{1} - \mathbf{L}^\top)\mathbf{L}\vec{u}_i + 2\beta\vec{\gamma}^i(\mathbf{I} - \mathbf{E}_i^\top)\vec{u}_i$$

where the constant parts are eliminated. So, using the kernel function $\mathcal{K}(\mathbf{X}, \mathbf{X})$, we re-formulate (9) for updating $\vec{u}_i$ as

$$\begin{aligned} \min_{\vec{u}_i} \quad & \beta\vec{u}_i^\top(\vec{\gamma}^i\vec{\gamma}^{i\top}(\mathcal{K} + \beta\mathbf{I}))\vec{u}_i \\ & +\vec{\gamma}^i\big[\alpha(\mathbf{1} - \mathbf{L}^\top)\mathbf{L} + 2\beta(\mathbf{I} - \mathbf{E}_i^\top) - 2\mathbf{E}_i^\top\mathcal{K}\big]\vec{u}_i \\ \text{s.t.} \quad & \|\Phi(\mathbf{X})\vec{u}_i\|_2^2 = 1, \; \|\vec{u}_i\|_0 \leq T_0 \; , \; u_{ij} \in \mathbb{R}^+ \;\; \forall j \end{aligned} \tag{13}$$

Similar to (11), the above framework has the non-negative quadratic form with the cardinality constraint $\|\vec{u}_i\|_0 \leq T_0$ and is a convex problem similar to (11). Consequently, its solution can be approximated using the NQP algorithm [16].

Note: When computing $\mathbf{E}_i$ to update $\vec{u}_i$, matrix $\mathbf{U}$ should be used with its recently updated columns to improve the convergence speed of the optimization loop. For example, in a successive update of columns in $\mathbf{U}$ starting from the 1st column, we compute $\mathbf{E}_i$ at iteration t by using matrix $\mathbf{U}$ as:

$$\mathbf{U} = \{\vec{u}_1^1, \vec{u}_2^2, ..., \vec{u}_{i-1}^{(t-1)}, \vec{u}_i^1, ..., \vec{u}_k^1\}$$

where $\vec{u}_i^t$ is the value of $\vec{u}_i$ at iteration t of the dictionary updating loop. Besides, directly after updating each $\vec{u}_i$ via NQP algorithm, it should be normalized as $\vec{u}_i \rightarrow \frac{\vec{u}_i}{\|\Phi(\mathbf{X})\vec{u}_i\|_2}$.

Algorithm 1 Confident Kernel Sparse Coding

Parameters: λ, T_0.
Input: Labels $\mathbf{L}$, kernel matrix $\mathcal{K}(\mathbf{X}, \mathbf{X})$.
Output: Sparse coefficients $\mathbf{\Gamma}$, discriminant dictionary $\mathbf{U}$.
Initialization: Computing β based on section 3.2.
repeat
  Updating $\mathbf{\Gamma}$ based on (11) using NQP
  Updating $\mathbf{U}$ based on (13) using NQP
until Convergence

4.3 Update of the Recall Phase $\vec{\gamma}$:

In order to reconstruct the test data $\vec{z}$, its corresponding sparse code $\vec{\gamma}$ is approximated via expanding (10) as follows

$$\begin{aligned} \min_{\vec{\gamma}} \quad & \vec{\gamma}^\top \mathbf{U}^\top \big[\mathcal{K} + \alpha \mathbf{L}^\top (\mathbf{1} - \mathbf{I})\mathbf{L} + \beta \mathbf{I}\big] \mathbf{U}\vec{\gamma} \\ & -2\mathcal{K}(\vec{z}, \mathbf{X})\mathbf{U}\vec{\gamma} \\ \text{s.t.} \quad & \|\vec{\gamma}\|_0 < T_0, \;\; \gamma_j \in \mathbb{R}^+ \;\; \forall j \end{aligned} \tag{14}$$

This optimization problem is convex based on the analysis provided in Sec. 3.2, and can be approximately solved by the NQP algorithm [16] similar to the update of $\mathbf{\Gamma}$ and $\mathbf{U}$. The algorithm's code will be available in an online repository[2].

5 Experiments

In order to evaluate our proposed confident kernel sparse coding algorithm, we carry out implementations on the following selection of sequential datasets.

- Cricket Umpire[17]: A collection of 180 instances of hand movement data related to 12 different types of cricket umpire's signal [17].
- Articulatory Words[18]: Recorded data of facial movements using EMA sensors while uttering 25 different words, containing 575 data samples [18].
- Schunk Dexterous [19]: Tactile data of robot grasping for 5 different objects with 10 samples per object. The dimensions are extracted based on the work of [20].
- UTKinect Actions [21]: 3D location of body joints recorded using Kinect device related to 9 different actions, containing 199 action instances in total.
- DynTex++: A large-scale Dynamic Texture dataset with 36 classes and 100 sequences per category [22].

For DynTex++, the kernel is computed based on [23]; however, for other datasets, we compute the Gaussian kernel

$$\mathcal{K}(\vec{x}_i, \vec{x}_j) = exp(-\mathcal{D}(\vec{x}_i, \vec{x}_j)^2/\delta)$$

where $\mathcal{D}(\vec{x}_i, \vec{x}_j)$ is the distance between a pair of $\{\vec{x}_i, \vec{x}_j\}$. The value of $\mathcal{D}(\vec{x}_i, \vec{x}_j)$ is computed using Dynamic Time Warping method [24], and δ is determined as the average of $\mathcal{D}(\vec{x}_i, \vec{x}_j)$ over all data points.

[2] https://github.com/bab-git/CKSC

Table 1: Parameter settings of CKSC regarding each dataset

Parameter	Cricket	Words	Schunk	UTKinect	DynTex++
T_0	6	8	5	7	15
λ	0.2	0.2	0.15	0.1	0.15

We compare our proposed method to the relevant baseline algorithms selected as K-KSVD [7], JKSRC [25], LC-NNKSC [15], LP-KSVD [5], KGPL [26], and EKDL [23], which are known as the most recent kernel-based discriminative sparse coding algorithms. The basis of our comparisons is the average classification accuracy for 10 randomized test/train selections, which shows the quality of the resulted discriminative representations for each method.

$$\text{accuracy} = \frac{\{\#\text{ correctly assigned labels}\}}{N} \times 100$$

Note: It is important to emphasize that the purpose of our discriminative sparse coding framework is to obtain a **sparse discriminant representation** of the data based on its pre-computed **kernel representation**. Therefore, instead of comparing the results to all the available top classifiers such as Deep Neural Networks and others, we only select the recent kernel-based alternatives which fit the above description.

Parameters Tuning: In order to tune the parameters λ and T_0 related to the optimization framework of CKSC in (4 , 5), we also perform 5-fold cross-validation. We carry out the same procedure for the baselines to find their optimal choice of parameters. The parameter k (dictionary size) is determined by choosing the *number of dictionary atoms per class*:

$$\{\#\text{ atoms per class}\} = \{\#\text{ classes}\} \times T_0$$

However, in practice as a working parameter setting for CKSC, we can choose a value around $\lambda = 0.1$. Parameter T_0 (and the dictionary size) depends on class distributions and complexity of the dataset. However, based on practical evidence having large values for T_0 does not improve the performance of CKSC and only increases the dictionary redundancy. Table 1 the chosen parameter values for CKSC algorithm regarding each dataset.

5.1 Classification Results

The classification accuracies of the implementations are reported in Table 2. According to the given results, our CKSC algorithm obtains the highest classification performance for all of the benchmarks, which shows that the designed frameworks of (4) and (5) provide better discriminative representations in comparison to other K-SRC algorithms.

CKSC, EKDL, LP-KSVD, and LC-NNKSC algorithms can be considered as the runner-up methods which have competitive classification accuracies. Their good performance is due to the embedded labeling information in their discriminant terms which improves their discriminative representations in contrast to K-KSVD, JSRC, and KGDL. Nevertheless, there is a variation in the comparison results of these methods. We can conclude that although they use different strategies to obtain a discriminant model, their recall model does not necessarily comply with their training model. In contrast, CKSC demonstrated that it has a more efficient embedding of the supervised information in both of the training and the recall framework.

LC-NNKSC uses a non-negative framework similar to the basis of CKSC's structure, and via comparing the results of LC-NNKSC to those of LP-KSVD and EKDL we observe that its non-negative framework obtains a competitive performance. Although LP-KSVD and EKDL employ extra objective terms in their models, this non-negative structure can achieve a similar outcome without the need to use such extra terms. Relevantly, CKSC benefits from this non-negative optimization framework as a basis for its confidence based model which leads to its superior performance compared to other baselines.

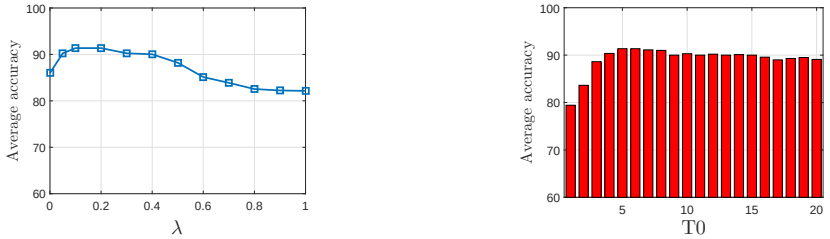
5.2 Sensitivity to the Parameter Settings

To study the sensitivity of CKSC to the parameter settings, we carry out experiments via changing the algorithm's parameters (λ, T_0). Implementing on Schunk dataset, we apply CKSC in 2 individual settings via changing one parameter throughout each experiment when the other one is fixed

Table 2: Average classification accuracies (%) $\pm$ standard deviations for the selected datasets.

Datasets	K-KSVD	JKSRC	LC-NNKSC	LP-KSVD	KGDL	EKDL	**CKSC**
Schunk	83.42±0.35	87.49±0.57	89.96±0.64	89.62±0.51	88.17±0.43	88.39±0.24	**91.42±0.34**
DynTex++	89.22±0.47	89.95±0.35	93.22±0.37	93.12±0.47	92.83±.31	93.51±0.46	**94.36±0.32**
Words	93.35±0.84	92.14±0.78	97.33±0.75	97.64±0.67	95.82±0.66	96.53±0.68	**98.82±0.69**
Cricket	75.12±1.54	78.14±1.38	83.33±1.12	83.33±0.95	82.78±1.07	84.25±0.95	**86.45±0.85**
UTKinect	80.26±0.35	81.47±0.42	84.12±0.32	84.93±0.25	83.51±0.36	84.18±0.24	**86.52±0.26**

The best result (**bold**) is according to a two-valued t-test at a 5% significance level.



(a) (b)

Figure 1: Sensitivity analysis of CKSC to hyper-parameters (a)λ and (b)T_0 for Schunk dataset.

to the value reported in Table 1. As it is observed from Figure 1a, a good choice for λ lies in the interval $[0.1, 0.4]$. However, the discriminative objective can outweigh the reconstruction part for values of λ close to 1, and it results in over-fitting and performance reductions.

Regarding the dictionary size, we increase T_0 from 1 to 20 with step-size 1 which changes the size of $\mathbf{U}$ in the range $[20, 400]$ with step-size 20 (average number of data samples per class). According to Figure 1b, for Schunk dataset, having T_0 between 4 and 8 keeps the performance of CKSC at an optimal level. As it is clear, small values of T_0 put a tight limit on the number of available atoms $\vec{u}_i$ for reconstruction purpose which reduces the accuracy of the method. On the other hand, larger values of T_0 increase dictionary redundancy and loosen up the sparseness bound on $\mathbf{\Gamma}$; nevertheless, NQP algorithm and the non-negativity constraints intrinsically incur sparse characteristics to $\mathbf{\Gamma}$ and $\mathbf{U}$ via combining only the most similar resources. Therefore, increasing T_0 does not dramatically degrades the performance of CKSC.

5.3 Complexity and Convergence of CKSC

In order to calculate the computational complexity of CKSC per iteration, we analyze the update of $\{\mathbf{\Gamma}, \mathbf{U}\}$ separately. In each iteration, $\mathbf{\Gamma}$ and $\mathbf{U}$ are optimized using the NQP algorithm which has the computational complexity of $\mathcal{O}(NT_0^2)$ [16], where T_0 and N are the sparsity limit and the number of training samples. Therefore, optimizing $\mathbf{\Gamma}$ and $\mathbf{U}$ cost $\mathcal{O}(kNT_0^2 + kN^2)$ and $\mathcal{O}(kNT_0^2 + kcN^2 + kN^3)$ respectively, in which c is the dictionary size.

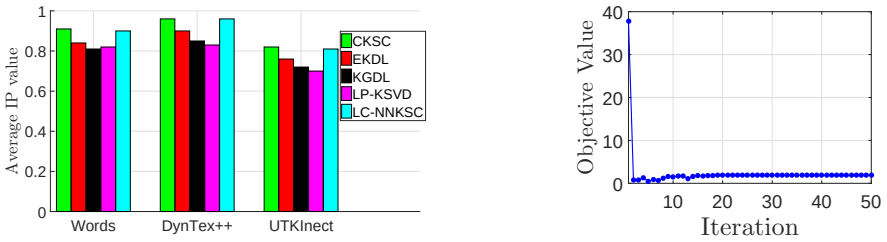
As shown in Figure 1b, practically we choose $T_0 \leq 10$, and also $c < k$. Therefore the total time complexity of each iteration is $\mathcal{O}(kN^3)$, in which, the dominant parts of the computational costs are dedicated to pre-optimization computations. Nevertheless, for datasets that N/c is relevantly large, the size of $\mathbf{U}$ should be chosen as $k \ll N$ in practice. Otherwise, it increases the redundancy in the dictionary without having any added-value. Therefore, in practice, the computational complexity of CKSC becomes $\mathcal{O}(N^3)$.

The optimization framework of CKSC in (9) is non-convex when considering $\{\mathbf{U}, \mathbf{\Gamma}\}$ together. However, each of the sub-problems defined in (11) and (13) are convex. Therefore, the alternating optimization scheme in Algorithm 1 converges in a limited number of steps.

5.4 Interpretability of the Dictionary

We define the interpretability measure IP_i for each $\vec{u}_i$ as

$$IP_i = \max_j (\vec{\rho}_j^\top \vec{u}_i) / (\vec{\mathbf{1}}^\top \mathbf{L} \vec{u}_i)$$



(a) (b)

Figure 2: (a) Average IP value for UTKinect, Words, and DaynTex++. (b) Convergence curve of CKSC for Cricket.

where $\vec{\mathbf{1}} \in \mathbb{R}^c$ is a vector of ones. IP_i becomes 1 if $\vec{u}_i$ uses data instances related only to one specific class. Figure 2a presents the IP value for the algorithms CKSC, EKDL, LC-NNKSC, KGDL, and LP-KSVD related to their implementations on the datasets Words, DynTex++, and UTKinect. Based on the results, CKSC and LC-NNKSC achieved the highest IP values as they use similar non-negative constraints in their training frameworks. Among others, EKDL presents better interpretability results due to the incoherency term it uses between the dictionary atoms $\vec{u}_i$. Putting the above results next to the classification accuracies (Table 2), we conclude that the CKSC algorithm learns highly interpretable dictionary atoms $\vec{u}_i$ while providing an efficient discriminative representation.

6 Conclusion

In this work, we presented a novel kernel-based discriminant sparse coding and dictionary learning framework, which relies on the discriminative confidence of each sparse codes in an individual class of data. Our CKSC algorithm incorporates the labeling information in training of the dictionary as well as in reconstruction of the test data in the kernel space, which increases the consistency between its trained and recall models. It also benefits from a non-negative framework which facilitates the discriminant reconstruction of the data points using the contributions taken from the most similar class of data. CKSC algorithm is presented and discussed comprehensively. Based on the empirical evaluations on time-series datasets, CKSC outperforms the relevant kernel-based sparse coding baselines due to higher classification performance based on the obtained representations. It also learns dictionaries with the more interpretable structures compared to other algorithms.

Acknowledgments

This research was supported by the Cluster of Excellence Cognitive Interaction Technology 'CITEC' (EXC 277) at Bielefeld University, which is funded by the German Research Foundation (DFG).

References

[1] B. Hosseini and B. Hammer, "Confident kernel sparse coding and dictionary learning," in *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2018, pp. 1031–1036.

[2] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, 2006.

[3] T. Kim, G. Shakhnarovich, and R. Urtasun, "Sparse coding for learning interpretable spatio-temporal primitives," in *NIPS'10*, 2010, pp. 1117–1125.

[4] J. Mairal, F. Bach, and J. Ponce, "Task-driven dictionary learning," *IEEE TPAMI*, vol. 34, no. 4, pp. 791–804, 2012.

[5] W. Liu, Z. Yu, M. Yang, L. Lu, and Y. Zou, "Joint kernel dictionary and classifier learning for sparse coding via locality preserving k-svd," in *Multimedia and Expo (ICME'15)*. IEEE, 2015, pp. 1–6.

[6] C. Zhang, J. Liu, Q. Tian, C. Xu, H. Lu, and S. Ma, "Image classification by non-negative sparse coding, low-rank and sparse decomposition," in *CVPR'11*. IEEE, 2011, pp. 1673–1680.

[7] H. V. Nguyen, V. M. Patel, N. M. Nasrabadi, and R. Chellappa, "Design of non-linear kernel dictionaries for object recognition," *IEEE Transactions on Image Processing*, vol. 22, no. 12, pp. 5123–5135, 2013.

[8] S. Bahrampour, N. M. Nasrabadi, A. Ray, and K. W. Jenkins, "Kernel task-driven dictionary learning for hyperspectral image classification," in *ICASSP'15*. IEEE, 2015, pp. 1324–1328.

[9] Y. Quan, Y. Xu, Y. Sun, and Y. Huang, "Supervised dictionary learning with multiple classifier integration," *Pattern Recognition*, vol. 55, pp. 247–260, 2016.

[10] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Royal Stat. Soc. Series B (Methodological)*, pp. 267–288, 1996.

[11] F. Bach, J. Ponce, G. Sapiro, J. Mairal, and A. Zisserman, "Learning discriminative dictionaries for local image analysis." CVPR, 2008.

[12] R. Jenatton, J. Mairal, G. Obozinski, and F. R. Bach, "Proximal methods for sparse hierarchical dictionary learning." in *ICML'10*, no. 2010. Citeseer, 2010, pp. 487–494.

[13] M. Yang, L. Zhang, X. Feng, and D. Zhang, "Fisher discrimination dictionary learning for sparse representation," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 543–550.

[14] S. Kong and D. Wang, "A dictionary learning approach for classification: separating the particularity and the commonality," in *European Conference on Computer Vision*. Springer, 2012, pp. 186–199.

[15] B. Hosseini, F. Hülsmann, M. Botsch, and B. Hammer, "Non-negative kernel sparse coding for the analysis of motion data," in *International Conference on Artificial Neural Networks (ICANN)*. Springer, 2016, pp. 506–514.

[16] B. Hosseini, F. Petitjean, F. G., and B. Hammer, "Confident kernel dictionary learning for discriminative representation of multivariate time-series," *under review article*, 2019.

[17] M. H. Ko, G. W. West, S. Venkatesh, and M. Kumar, "Online context recognition in multisensor systems using dynamic time warping," in *ISSNIP'05*. IEEE, 2005, pp. 283–288.

[18] J. Wang, A. Samal, and J. Green, "Preliminary test of a real-time, interactive silent speech interface based on electromagnetic articulograph," in *SLPAT'14*, 2014, pp. 38–45.

[19] A. Drimus, G. Kootstra, A. Bilberg, and D. Kragic, "Design of a flexible tactile sensor for classification of rigid and deformable objects," *Robotics and Autonomous Systems*, vol. 62, no. 1, pp. 3–15, 2014.

[20] M. Madry, L. Bo, D. Kragic, and D. Fox, "St-hmp: Unsupervised spatio-temporal feature learning for tactile data," in *ICRA'14*. IEEE, 2014, pp. 2262–2269.

[21] L. Xia, C.-C. Chen, and J. Aggarwal, "View invariant human action recognition using histograms of 3d joints," in *CVPRW'12 Workshops*. IEEE, 2012, pp. 20–27.

[22] B. Ghanem and N. Ahuja, "Maximum margin distance learning for dynamic texture recognition," in *ECCV'10*. Springer, 2010, pp. 223–236.

[23] Y. Quan, C. Bao, and H. Ji, "Equiangular kernel dictionary learning with applications to dynamic texture analysis," in *CVPR'16*, 2016, pp. 308–316.

[24] K. Adistambha, C. Ritz, and I. Burnett, "Motion classification using dynamic time warping," in *MMSP'08 Workshop*, Oct 2008, pp. 622–627.

[25] H. Liu, D. Guo, and F. Sun, "Object recognition using tactile measurements: Kernel sparse coding methods," *IEEE Transactions on Instrumentation and Measurement*, vol. 65, no. 3, pp. 656–665, 2016.

[26] M. Harandi, C. Sanderson, C. Shen, and B. Lovell, "Dictionary learning and sparse coding on grassmann manifolds: An extrinsic solution," in *ICCV'13*. IEEE, 2013, pp. 3120–3127.