Cogn Comput
DOI 10.1007/s12559-016-9438-0

A Semi-Supervised Predictive Sparse Decomposition Based on Task-Driven Dictionary Learning

Le Lv[1] · Dongbin Zhao[1,2] · Qingqiong Deng[3]

Received: 3 November 2015 / Accepted: 25 October 2016

Abstract In feature learning field, many methods are inspired by advances in neuroscience. Among them, neural network and sparse coding have been broadly studied. Predictive sparse decomposition (PSD) is a practical variant of these two methods. It trains a neural network to estimate the sparse codes. After training, the neural network is fine-tuned to achieve higher performance on object recognition tasks. It is widely believed that introducing discriminative information can make the features more useful for classification task. Hence, in this work, we propose applying the task-driven dictionary learning framework to the PSD and demonstrate that this new model can be optimized by the stochastic gradient descent (SGD) algorithm. Before our work, the semi-supervised auto-encoder framework has already been proposed to guide neural network to extract discriminative representations. But it does not improve the classification performance of neural network. In the experiments, we compare the proposed method with the semi-supervised auto-encoder method. The performance of PSD is used as the baseline for these two methods. On the MNIST and USPS datasets, our method can generate more discriminative and predictable sparse codes than other methods. Furthermore, the recognition accuracy of neural network can be improved.

Keywords Semi-supervised learning · Predictive sparse decomposition · Neural networks · Dictionary learning

Introduction

Inspired by the research on neuroscience, various feature learning methods have been proposed [1, 2]. Among them, neural network and sparse coding have been broadly studied. These two methods are inspired by the mechanisms of human visual cortex. PSD proposed by Kavukcuoglu et al. combines these two methods [3]. It is an unsupervised feature learning method. The sparse coding learns a set of abstract basis functions and represents the input signals as sparse linear combinations of these basis functions. The neural network is trained to estimate the informative sparse codes and fine-tuned to achieve higher recognition accuracy. Many variants of PSD have been proposed [4–6].

Unsupervised feature learning methods can extract abstract representations [7]. However, their performance

This work is supported by National Natural Science Foundation of China (NSFC) under Grants No. 61273136, No. 61573353 and No. 61533017.

✉ Dongbin Zhao
dongbin.zhao@ia.ac.cn

Le Lv
lvle2012@ia.ac.cn

Qingqiong Deng
qqdeng@bnu.edu.cn

[1] Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China

[2] University of Chinese Academy of Sciences, Beijing, China

[3] College of Information Science and Technology, Beijing Normal University, Beijing, 100875, China

Published online: 18 November 2016

contributions to the final recognition task are not understood clearly. It is widely believed that introducing discriminative information can make the representations more useful for classification task. Hence, to improve the recognition accuracy of neural network, the semi-supervised auto-encoder framework has been proposed [8]. However, a number of experiments show that this method does not improve the classification performance significantly. Bengio et al. explain that the introduction of supervised guidance is greedy. It may discard some of the information about the target.

Motivated by this problem, we propose a new semi-supervised learning method of neural network. We adopt the task-driven dictionary learning framework [9] to the PSD algorithm and demonstrate that the new model can be trained by SGD algorithm. The learned dictionary of our method is useful for both classification and reconstruction. Furthermore, the sparse codes are discriminative and representative. They provide more information to guide the neural network and can be estimated more accurately. Hence, the neural network can achieve higher recognition accuracy. For the convenience of discussion, in the following of this paper, our method is referred as the task-driven PSD and the early proposed semi-supervised auto-encoder method is referred as the semi-supervised PSD.

The rest of this paper is organized as follows: Section "Predictive Sparse Decomposition" presents the background of PSD and semi-supervised PSD algorithms. Section "Task-Driven Predictive Sparse Decomposition" presents our task-driven PSD algorithm and explains the corresponding optimization algorithm. Section "Experiments" is devoted to several experiments on the MNIST and USPS datasets. In Section "Discussion," we discuss some and discuss future work.

Predictive Sparse Decomposition

Sparse Coding

Sparse coding is a widely used approach in the computer vision and image processing fields. A signal $x \in R^m$ can be represented by a linear combination of an over-complete set of prototype signal-atoms $\{d_j\}_{j=1}^{p}$. These basis vectors form the dictionary matrix $D \in R^{m\times p}$. The representation coefficients of x are denoted by the vector $\alpha \in R^p$. The representation can be either exact $x = D\alpha$ or approximate $\|x - D\alpha\| \leq \varepsilon$. Due to the over-completeness of D, infinitely many solutions are available for the representation. Many applications can benefit from sparse representations. Hence, the solution is constrained to have the fewest number of nonzero coefficients. The problem can be formulated as:

$$\min_{\alpha} \|\alpha\|_0 \quad \text{s.t.} \quad x = D\alpha \quad \text{or}$$
$$\min_{\alpha} \|\alpha\|_0 \quad \text{s.t.} \quad \|x - D\alpha\|_2 \leq \varepsilon$$

where $\|\cdot\|_0$ is the L_0 norm. A combinatorial search can be used to find the optimal solution. However, it is intractable in high-dimensional spaces. Mallet et al. introduced the matching pursuit method to approximate the solution greedily [10]. An alternative method is known as Lasso [11] or basis pursuit [12]. It makes a convex relaxation by replacing the L_0 norm to the L_1 norm. The lasso problem can be written as:

$$\min_{\alpha} \frac{1}{2}\|x - D\alpha\|_2^2 + \lambda_1\|\alpha\|_1 \tag{1}$$

where the λ_1 is the regularization coefficient to control the sparsity of α.

In many situations, the Lasso is appealing. However, in the over-complete case, the Lasso selects at most m atoms and is not well defined. Hence, Zou et al. defined the elastic net criterion [13]:

$$\min_{\alpha} \frac{1}{2}\|x - D\alpha\|_2^2 + \lambda_1\|\alpha\|_1 + \frac{\lambda_2}{2}\|\alpha\|_2^2$$

The elastic net can select all p atoms. At the same time, the strict convexity of elastic net guarantees the grouping and decorrelation effect of solutions.

To solve the sparse coding problem, many efficient algorithms have been studied such as coordinate descent (CD) [14] or least angle regression (LARS-Lasso) [15] methods. However, these algorithms are computationally expensive because all of them involve an iterative minimization process.

Dictionary Learning

In the early studies, the dictionary matrix is pre-specified by a set of functions, for example wavelets [16], curvelets [17], and contourlets [18]. An appropriate dictionary must be chosen for a specific question, otherwise the representation will not be sparse. Hence, learning a dictionary to fit the given training set is appealing [19, 20].

Given a finite training set $X = [x_1, ..., x_n] \in R^{m\times n}$, a dictionary matrix D is found by minimizing the empirical cost function:

$$J_u(D) = \frac{1}{n}\sum_{t=1}^{n} l_u(x_t, D) \tag{2}$$

where $l_u(x, D)$ is the sparse coding loss. (2) means that if D is suitable to represent the signals in X sparsely, the empirical cost should be small [9]. In order to prevent trivial solutions where D is arbitrarily large and α is arbitrarily small, the atoms of dictionary are constrained to have L_2 norm not greater than 1. The convex set of matrices satisfying this constraint is defined as:

$$\Omega = \{D \in R^{m\times n} \quad \text{s.t} \quad \forall j \in \{1, ..., p\}, \|d_j\|_2 \leq 1\}.$$

The dictionary learning problem is a joint optimization problem with respect to the dictionary D and the sparse representations $[\alpha_1, ..., \alpha_n] \in R^{p\times n}$. This is not joint convex, but convex with respect to each of the two variables when the other one is fixed [21]. After training, we will use the learned dictionary to calculate the representation of new example x.

Predictive Sparse Decomposition

As discussed above, the dictionary can be adapted for special dataset flexibly. And the sparse codes are robust to noise in the input signals. Due to these advantages, the sparse coding is adopted to regularize the pre-training of neural networks. The PSD can be formulated as:

$$J_{psd}(D, P_f) = \frac{1}{n}\sum_{t=1}^{n} l_{psd}(x_t, D, P_f)$$

$$l_{psd}(x, D, P_f) = \min_{\alpha} \frac{1}{2}\|x - D\alpha\|_2^2 + \lambda_1\|\alpha\|_1 + \frac{\lambda_2}{2}\|\alpha - f(x, P_f)\|_2^2 \tag{3}$$

where $f(x, P_f)$ is the single layer neural networks. It can be written as:

$$f(x, P_f) = G\tanh(Wx + b)$$

where $W \in R^{p\times m}$ is a filter matrix, $b \in R^p$ is a bias vector, tanh is the hyperbolic tangent function, and $G \in R^{p\times p}$ is a diagonal matrix of gain coefficients. For the sake of convenience in writing, we use $P_f = \{G, W, b\}$ to denote all parameters.

From the view of dictionary learning, the PSD can be seen as a kind of regularizer that forces the sparse codes to be nearly predictable by a smooth encoder. λ_2 is used to control the tradeoff between the prediction accuracy and the sparse coding loss. After the pre-training, the neural network can estimate the sparse codes and avoid the costly inference operation. The dictionary learning problem is not convex. Therefore, if multiple local optimal dictionary matrices exist, the PSD criterion will drive the system towards producing easily predictable representations. In [3], the stochastic gradient descent (SGD) algorithm is used to train the dictionary matrix and regressor parameters.

Semi-supervised Predictive Sparse Decomposition

To extract discriminative features, Bengio et al. propose the semi-supervised auto-encoder method. A soft-max layer is added to the encoder function and back propagates the classification error to it. Hence, the semi-supervised PSD is formulated as:

$$\min_{D, P_f, W_c} \mu J_{psd}(D, P_f) + (1-\mu)J_s(W_c, P_f) \tag{4}$$

where μ is the tradeoff parameter and J_s is the classification loss. For the multi-class classification task, the label associated with x_t is denoted by a 1-of-k vector y_t. The label space Ψ contains k discrete classes. The label set is denoted by $Y = [y_1, ..., y_n]$. J_s can be chosen as the soft-max loss. It is computed as:

$$J_s(W_c) = -\frac{1}{n}\sum_{t=1}^{n} l_s(y_t, W_c, f_t) \tag{5}$$

$$l_s(y_t, W_c, f_t) = \sum_{i=1}^{k} y_{ti}\ln(q_{ti}) \tag{6}$$

$$q_{ti} = \frac{\exp(W_{c,i}^T f_t)}{\sum_{j=1}^{k}\exp(W_{c,j}^T f_t)} \tag{7}$$

where f_t is the abbreviation for $f(x_t, P_f)$, W_c are the parameters of soft-max classifier, $W_{c,j}^T$ is the j'th row vector of W_c, and y_{ti}, q_{ti} are the i'th components for the label vector y_t and the predicted probability vector q_t.

The learning procedure of semi-supervised PSD is summarized in Algorithm 1. Firstly, in the initialization, we need to determine the hyper-parameters including the L_1 regularization coefficient λ_1, the regression regularization coefficient λ_2, the tradeoff parameter μ, the number of basis vectors, the number of iterations T, and the parameters ρ, a, r which control the decline of learning rate. In each iteration, the algorithm use the previous updated dictionary matrix D and the regressor parameters P_f to compute the approximate sparse representation f_t and the optimal sparse representation $\alpha^\star$ of x_t. Here, the CD algorithm is used to infer $\alpha^\star$. f_t is used as the initial value for the CD algorithm. Then $\alpha^\star$ is fixed. The gradient of J_s with respect to the W_c, P_f and the gradient of J_{psd} with respect to D, P_f are computed. The learning rate is computed by $\rho_t = \rho/(1 + at)^r$. Finally, W_c, D, and P_f are updated. Because the dictionary matrix D is constrained in Ω, an additional orthogonal projection operation should be done.

Algorithm 1 Stochastic gradient descent algorithm for semi-supervised PSD

```
Parameters initialization:
regularization parameters $\lambda_1, \lambda_2$, tradeoff parameter $\mu$, dictionary matrix $D$, regressor parameters $P_f$, classifier parameters $W_c$, total number of iterations $T$, and learning rate parameters $\rho, a, r$.
while t=1:T do
    Draw $(x_t, y_t)$ from training set $(X, Y)$.
    Sparse coding: compute $\alpha^\star$ using the CD algorithm.
    $\arg\min_{\alpha} \frac{1}{2}\|x - D\alpha\|_2^2 + \lambda_1\|\alpha\|_1 + \frac{\lambda_2}{2}\|\alpha - f(x, P_f)\|_2^2$
    Compute the gradient of $W_c$:
    $\nabla W_c = (1 - \mu)\frac{\partial l_s}{\partial W_c}$.
    Compute the gradient of $D$:
    $\nabla D = \mu(D\alpha^\star - x_t)\alpha^{\star T}$
    Compute the gradient of $P_f$ using back-propagation algorithm.
    $\nabla P_f = \mu\frac{\partial l_{psd}}{\partial P_f} + (1 - \mu)\frac{\partial l_s}{\partial P_f}$
    Get the learning rate $\rho_t = \rho/(1 + at)^r$.
    Update the parameters by a projected gradient step.
end while
Return $W_c$, $D$, and $P_f$.
```

Task-Driven Predictive Sparse Decomposition

It is well known that the discriminative power of representations greatly influences the recognition accuracy. Intuitively, the semi-supervised auto-encoders should be better than the unsupervised pre-training methods. However, the experiment results in [8] are unsatisfactory. Bengio et al. explain that the introduction of supervised guidance may be too greedy. Some of the information about the target is discarded. In [22], Ba et al. propose the mimic learning methods. They use the informative intermediate features extracted by deep nets to guide the training of shallow networks. The mimic learning method is similar to the PSD. Both of them train the neural networks to regress the representations provided by other models. Inspired by the mimic learning, we expect to use a more delicate model to learn the semi-supervised task and then use its informative intermediate representations to guide the neural networks. There are many variants of the sparse coding. Among them, the task-driven framework [9] is one of the most outstanding discriminative sparse coding methods. Hence, we combine it with the PSD. Within this framework, the optimal sparse representations are used for the classification task. We show that our task-driven PSD can also be trained by the SGD algorithm. Finally, we experimentally verify that the proposed method can improve the pre-training process and the fine-tuned neural network can achieve higher recognition performance.

Basic Formulation

In our task-driven PSD, the classification depends on the optimal sparse representation $\alpha^\star$. The soft-max loss in (5) is still used here. This will be a joint optimization of D, W_c, and P_f. The formulation is written as:

$$\min_{D, P_f, W_c} (1 - \mu)J_s(W_c, D, P_f) + \mu J_{psd}(D, P_f)$$

$$J_s(W_c, D, P_f) = E_{y,x}[l_s(y, W_c, \alpha^\star(x, D, P_f))]$$

$$\alpha^\star(x, D, P_f) = \arg\min_{\alpha} \frac{1}{2}\|x - D\alpha\|_2^2 + \lambda_1\|\alpha\|_1 + \frac{\lambda_2}{2}\|\alpha - f(x, P_f)\|_2^2$$

The expectation is taken relative to the probability distribution $p(x, y)$. To train D and P_f in a supervised way, $\alpha^\star$ should be differentiable with respect to D and f.

Differentiability Analysis

Before the differentiability analysis of $\alpha^\star$, we will discuss some assumptions. We assume that the probability distribution $p(x, y)$ has a compact support $K_x \times K_y \subseteq R^m \times \Psi$. This assumption is reasonable because the data acquired by sensor is bounded. Additionally, for all labels $y \in \Psi$, $p(\cdot, y)$ is continuous and $l_s(y, \cdot)$ is twice continuously differentiable.

To study the differentiability of $\alpha^\star$, it is essential to prove that $\alpha^\star$ can be expressed as a function of x, D, and f. Then the continuity of the function $\alpha^\star(x, D, f)$ should be discussed. On the basis of continuity, we can further study the differentiability. At last, according to the sub-gradient optimality conditions given by Fuchs [23], we can derive the explicit function $\alpha^\star$ with respect to x, D, f and compute the gradient of D and P_f.

Theorem 1 *With fixed $D \in \Omega$ and P_f, for any $x \in R^m$, the sparse representation $\alpha^\star$ inferred by the PSD criterion is unique.*

Proof We will do an equivalent algebraic transformation on the PSD criterion. We define an augmented data and dictionary

$$\tilde{x} = \begin{bmatrix} x \\ \sqrt{\lambda_2} f(x, P_f) \end{bmatrix} \quad \text{and}$$

$$\tilde{D} = \frac{1}{\sqrt{1+\lambda_2}} \begin{bmatrix} x \\ \sqrt{\lambda_2} f(x, P_f) \end{bmatrix}$$

Then, the PSD criterion can be rewritten as:

$$\min_{\alpha} \frac{1}{2}\|\tilde{x} - \tilde{D}\tilde{\alpha}\|_2^2 + \gamma\|\tilde{\alpha}\|_1 \tag{8}$$

where $\tilde{\alpha} = \sqrt{1+\lambda_2}\alpha$ and $\gamma = \lambda_1/\sqrt{1+\lambda_2}$.

This transformation makes the PSD have a Lasso form. Obviously, the augmented dictionary matrix $\tilde{D}$ is column full rank. This makes the problem strictly convex. Hence, when $D \in \Omega$ and P_f are fixed, the optimal solution $\tilde{\alpha}^\star$ for any $x \in R^m$ is unique. So does the solution of PSD. □

According to Theorem 1, we can determine that there exists a function relationship between the optimal solution and the parameters of PSD. The classical optimality conditions for the Lasso will underlie our subsequent discussions. Hence, we briefly recall it here.

Lemma 1 (optimality conditions of the Lasso) *For $x \in K$ and $D \in \Omega$, $\alpha^\star \in R^p$ is a solution of (1) if and only if*

$$\begin{aligned} &d_j^T(x - D\alpha^\star) = \lambda_1 s_j \quad \textit{if} \alpha^\star \neq 0,\\ &|d_j^T(x - D\alpha^\star)| \leq \lambda_1 \quad \textit{otherwise.} \end{aligned}$$

where $\alpha_j^\star$, $j \in \{1, .., p\}$ is the j'th component of $\alpha^\star$ and $s_j = sign(\alpha_j^\star)$

Applying Lemma 1 to (8), we can obtain the optimality conditions of PSD.

$$\begin{aligned} &d_j^T(x - D\alpha^\star) + \lambda_2(f_j - \alpha_j^\star) = \lambda_1 s_j && \text{if } \alpha^\star \neq 0,\\ &|d_j^T(x - D\alpha^\star) + \lambda_2(f_j - \alpha_j^\star)| \leq \lambda_1 && \text{otherwise.} \end{aligned}$$

Due to the compactness of x and the continuity of the regressor f, the range of f is also compact. We denote the range of f as $K_f \subseteq R^p$.

Theorem 2 *The function $\alpha^\star$ is uniformly Lipschitz on $K_x \times \Omega \times K_f$.*

Proof In (9), $l(x, D, f, \alpha)$ is a continuous function.

$$l(x, D, f, \alpha) = \frac{1}{2}\|x - D\alpha\|_2^2 + \lambda_1\|\alpha\|_1 + \frac{\lambda_2}{2}\|\alpha - f\|_2^2 \tag{9}$$

For $x \in R^m$, $D \in \Omega$, and $f \in R^p$, $\alpha^\star(x, D, f)$ is the unique optimal solution. These imply that $\alpha^\star$ is continuous in x, D, f.

Since the quantity in (10) is continuous in x, D, and f,

$$d_j^T(x - D\alpha^\star) + \lambda_2(f_j - \alpha_j^\star) \tag{10}$$

there exists an open neighborhood V around (x, D, f) such that, if (x, D, f) satisfies (11), then $\forall(x', D', f') \in V$ also satisfies (11).

$$|d_j^T(x - D\alpha^\star) + \lambda_2(f_j - \alpha_j^\star)| < 0 \tag{11}$$

In Theorem 1, we have explained that $l(x, D, f, \cdot)$ is strictly convex. The Hessian matrix is $D^T D + \lambda_2 I$. Obviously, the smallest eigenvalue of Hessian matrix is not less than λ_2. Hence, $\forall(x', D', f') \in V$, we have

$$l(x, D, f, \alpha'^\star) - l(x, D, f, \alpha^\star) \geq \lambda_2\|\alpha'^\star - \alpha^\star\|_2^2 \tag{12}$$

Moreover, it is easy to show that (13) is Lipschitz with constant (14).

$$\tilde{l}(\cdot) = l(x, D, f, \cdot) - l(x', D', f', \cdot) \tag{13}$$

$$c = c_1\|x - x'\|_2 + c_2\|D - D'\|_F + c_3\|f - f'\|_2 \tag{14}$$

where c_1, c_2, c_3 are constants independent of x, x', D, D', f, f'.

Since $\alpha^\star$ is the optimal solution of $l(x, D, f, \cdot)$ and $\alpha'^\star$ is the optimal solution of $l(x', D', f', \cdot)$, we have

$$\begin{aligned} l(x, D, f, \alpha'^\star) \; - \; l(x, D, f, \alpha^\star) \geq 0\\ l(x', D', f', \alpha^\star) \; - \; l(x', D', f', \alpha'^\star) \geq 0 \end{aligned}$$

On the basis of (12), (15), and (16),

$$|\tilde{l}(\alpha'^\star) - \tilde{l}(\alpha^\star)| \leq c\|\alpha'^\star - \alpha^\star\|_2 \tag{15}$$

$$|\tilde{l}(\alpha'^\star) - \tilde{l}(\alpha^\star)| \geq l(x, D, f, \alpha'^\star) - l(x, D, f, \alpha^\star) \tag{16}$$

we can achieve that

$$\|\alpha'^\star - \alpha^\star\|_2 \leq \frac{1}{\lambda_2}(c_1\|x - x'\|_2 + c_2\|D - D'\|_F + c_3\|f - f'\|_2)$$

Therefore, $\alpha^\star$ is locally Lipschitz. Since $K_x \times \Omega \times K_f$ is compact, $\alpha^\star$ is uniformly Lipschitz. □

The proof for the continuity and uniformly Lipschitz of $\alpha^\star$ doesn't consider the regressor form of f. It is seen as a vector of independent variables. Our discussion is restricted in an open neighborhood of (x, D, f). Undoubtedly, if f is a continuous function in x, the conclusion is still true. In the next theorem, we will carry on this convention.

Theorem 3 *if $(x, D, f) \in K_x \times \Omega \times K_f$ and $\alpha^\star$ satisfies*

$$\begin{aligned} &s_j\alpha_j^\star \geq \varepsilon && \textit{if } \alpha_j^\star > 0,\\ &|d_j^T(x - D\alpha^\star) + \lambda_2(f_j - \alpha_j^\star)| \leq \lambda_1 - \varepsilon && \textit{otherwise.} \end{aligned}$$

then there exists a neighborhood of (x, D, f) where $\alpha^\star$ is twice continuously differentiable.

Proof Since $\alpha^\star(x, D, f)$ is uniformly Lipschitz, $s_j\alpha_j^\star$ and $d_j^T(x - D\alpha^\star) + \lambda_2(f_j - \alpha_j^\star)$, $\forall j \in \{1, ..., p\}$, are also

Lipschitz. There exists $\kappa > 0$ independent of x, D, f. If (x', D', f') satisfies

$$\|x - x'\|_2 \leq \kappa\varepsilon, \|D - D'\|_F \leq \kappa\varepsilon, \|f - f'\|_2 \leq \kappa\varepsilon \quad (17)$$

then, $\forall j \in \{1, ..., p\}$, we have

$$\begin{aligned} &s_j \alpha_j'^{\star} \geq \frac{\varepsilon}{2} && \text{if } s_j \neq 0, \\ &|d_j'^T (x' - D'\alpha'^{\star}) + \lambda_2 (f_j' - \alpha_j'^{\star})| \leq \lambda_1 - \frac{\varepsilon}{2} && \text{otherwise.} \end{aligned}$$

where $\alpha'^{\star}$ is shorthand for $\alpha^{\star}(x', D', f')$. Obviously, $\alpha'^{\star}$ has the same signs as $\alpha^{\star}$. Therefore, for any (x', D', f') satisfying (17), the signs of $\alpha^{\star}$ are stable. According to the optimality conditions of PSD, the nonzero coefficients of $\alpha^{\star}$ are given by the closed form:

$$\alpha_{\Lambda}^{\star} = (D_{\Lambda}^T D_{\Lambda} + \lambda_2 I)^{-1} (D_{\Lambda}^T x - \lambda_1 s_{\Lambda} + \lambda_2 f_{\Lambda}) \quad (18)$$

where $\Lambda = \{j \in \{1, ..., p\} \quad \text{s.t.} \quad \alpha_j^{\star} \neq 0\}$ is the active set. Hence, $\alpha^{\star}$ is locally twice differentiable. □

Theorem 4 *J_s is differentiable.*

Proof Apparently, J_s is differentiable in W_c. We will focus on proving the differentiability of J_s with respect to D. This proof can also be applied to f.

Since l_s is twice differentiable and $\alpha^{\star}$ is uniformly Lipschitz, by the Tayler theorem, we can derive that

$$\begin{aligned} &J_s(D', W_c, P_f) - J_s(D, W_c, P_f) \\ &\quad = E_{y,x,f}[l_s(y, W_c, \alpha'^{\star}) - l_s(y, W_c, \alpha^{\star})] \\ &\quad = E_{y,x,f}[\nabla_{\alpha} l_s^T (\alpha'^{\star} - \alpha^{\star})] \\ &\qquad + O(\|D' - D\|_F^2) \end{aligned}$$

where $\alpha'^{\star}$ is the shorthand for $\alpha^{\star}(x, D', f)$.

For $D \in \Omega$, a positive scalar ε determines a subset $K(D, \varepsilon) \subseteq K_x \times K_f$ of (x, f) satisfying Theorem 3. Then we have

$$\begin{aligned} &J_s(D', W_c, P_f) - J_s(D, W_c, P_f) \\ &\quad = \iint_{(x,f)\in K(D,\varepsilon)} \sum_{y\in\Psi} \nabla_{\alpha} l_s^T (\alpha'^{\star} - \alpha^{\star}) p(x, f, y) dx df \\ &\qquad + \iint_{(x,f)\in K^c(D,\varepsilon)} \sum_{y\in\Psi} \nabla_{\alpha} l_s^T (\alpha'^{\star} - \alpha^{\star}) p(x, f, y) dx df \\ &\qquad + O(\|D' - D\|_F^2) \end{aligned}$$

With the constant κ in Theorem 3, we can easily show that

$$P(K_x \times K_f - K(D, \frac{\|D' - D\|_F}{\kappa})) = O(\|D' - D\|_F)$$

At the same time, $\alpha^{\star}$ is uniformly Lipschitz, so we have

$$\nabla_{\alpha} l_s^T (\alpha'^{\star} - \alpha^{\star}) = O(\|D' - D\|_F)$$

Hence,

$$\begin{aligned} &J_s(D', W_c, P_f) - J_s(D, W_c, P_f) \\ &\quad = \iint_{(x,f)\in K(D,\varepsilon)} \sum_{y\in\Psi} \nabla_{\alpha} l_s^T (\alpha'^{\star} - \alpha^{\star}) \\ &\qquad \times p(x, f, y) dx df + O(\|D' - D\|_F^2) \end{aligned}$$

This shows that J_s is differentiable with respect to D. □

Task-Driven Predictive Sparse Decomposition

In the above section, we have proved the differentiability of $\alpha^{\star}(x, D, P_f)$. Hence, D, P_f can be trained by minimizing the classification loss. Our task-driven PSD algorithm is shown in Algorithm 2. In this algorithm, the initialization of parameters is similar to the semi-supervised PSD algorithm. We can divide each iteration of the algorithm into the forward and backward parts.

In the forward part, we need infer the sparse code $\alpha^{\star}$ of x_t and the active set Λ of $\alpha^{\star}$. Instead of the CD algorithm, the LARS-Lasso algorithm is used here. It is because the later gradient calculation involves the inverse of $D_{\Lambda}^T D_{\Lambda} + \lambda_2 I$. The LARS-Lasso algorithm containing a Cholesky factorization will be efficient. By the sparse code, the classification loss l_s is calculated.

In the backward part, the gradient of J_s and J_{psd} with respect to W_c, D, and P_f is computed. The gradient computation of J_{psd} is similar to Algorithm 1. Here, we will focus on the gradient computation of J_s. We first compute the gradient $\nabla_{W_c} l_s(y_t, \alpha^{\star}, W_c)$, $\nabla_{\alpha^{\star}} l_s(y_t, \alpha^{\star}, W_c)$. From the proof of Theorem 3, we know that the active set is stable when D and f change small. It makes the derivatives of D and f corresponding to the inactive set to be zero. Hence, irrelevant atom vectors and hidden units are dropped automatically. $\alpha^{\star}$ is the function of D and f. According to (18), we can derive the relevant derivatives. Then, the derivation chain rule can give the gradient of l_s with respect to D and f. For the convenience of writing, we introduce the intermediate variable $\beta^{\star}$:

$$\beta_{\Lambda}^{\star} = (D_{\Lambda}^T D_{\Lambda} + \lambda_2 I)^{-1} \nabla_{\alpha_{\Lambda}} l_s(y_t, \alpha^{\star}, W_c), \quad \beta_{\Lambda^c}^{\star} = 0$$

The gradient of l_s relative to D and f can be written as:

$$\nabla_D l_s = -D(\beta^{\star}\alpha^{\star T} + \alpha^{\star}\beta^{\star T}) + x_t \beta^{\star T}$$

$$\nabla_{f_{\Lambda}} l_s = \lambda_2 \beta_{\Lambda}^{\star}, \quad \nabla_{f_{\Lambda^c}} l_s = 0$$

With $\nabla_f l_s$, the gradient $\nabla_{P_f} l_s$ can be computed by back propagation algorithm. Then we weighted sum the gradient

Algorithm 2 Stochastic gradient descent algorithm for task-driven PSD

```
Parameters initialization:
regularization parameters $\lambda_1, \lambda_2$, tradeoff parameter $\mu$, dictionary matrix $D$, regressor parameters $P_f$, classifier parameters $W_c$, total number of iterations $T$, and learning rate parameters $\rho, a, r$.
while t=1:T do
    Draw $(x_t, y_t)$ from training set $(X, Y)$.
    Sparse coding: compute $\alpha^\star$ using the LARS-Lasso algorithm.
    Compute the active set:
        $\Lambda = \{j \in \{1, ..., p\} \quad \text{s.t.} \quad \alpha_j^\star \neq 0\}$.
    Compute $\nabla_{W_c} l_s(y_t, \alpha^\star, W_c)$.
    Compute $\beta^\star$:
        $\beta_\Lambda^\star = (D_\Lambda^T D_\Lambda + \lambda_2 I)^{-1} \nabla_{\alpha_\Lambda} l_s(y_t, \alpha^\star, W_c), \quad \beta_{\Lambda^c}^\star = 0$
    Compute the gradient of $l_s$ relative to $D$, $f$.
        $\nabla_D l_s = -D(\beta^\star \alpha^{\star T} + \alpha^\star \beta^{\star T}) + x_t \beta^{\star T}$
        $\nabla_{f_\Lambda} l_s = \lambda_2 \beta_\Lambda^\star, \quad \nabla_{f_{\Lambda^c}} l_s = 0$
    Compute the gradient of $l_s$ relative to $P_f$.
    Compute the gradient of $l_{psd}$, and weighted sum with the gradient of $l_s$.
    Compute the learning rate $\rho_t = \rho/(1 + at)^r$.
    Update the parameters by a projected gradient step.
end while
Return.
```

of supervised part and unsupervised part and update the parameters.

Our task-driven PSD is different from the early proposed semi-supervised PSD algorithm. It makes both the dictionary and neural network perceive discriminative information and generate more useful features for recognition task. The sparsity of representation is used to drop the irrelevant features.

Experiments

In this work, we test our proposed method on the MNIST and USPS datasets. Both of them are the database of handwritten digits. The MNIST dataset consists of 70,000 images. We show its examples in Fig. 1a. The size of images in MNIST is 28 × 28. The dataset is partitioned into 60,000 training examples and 10,000 testing examples. Most of related work take out 10,000 images from the training set to validate and choose the best hyper-parameters. In the USPS dataset, there are 7291 training examples and 2007 testing examples. In Fig. 1b, the examples in USPS are shown. The digit images have been deslanted and size normalized, resulting in 16×16 grayscale images. For the USPS dataset, we keep 10 % of the training data for validation.

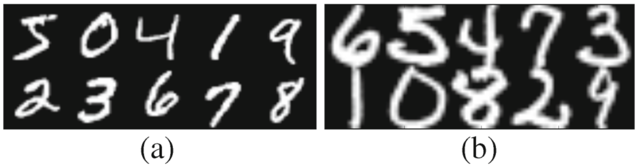
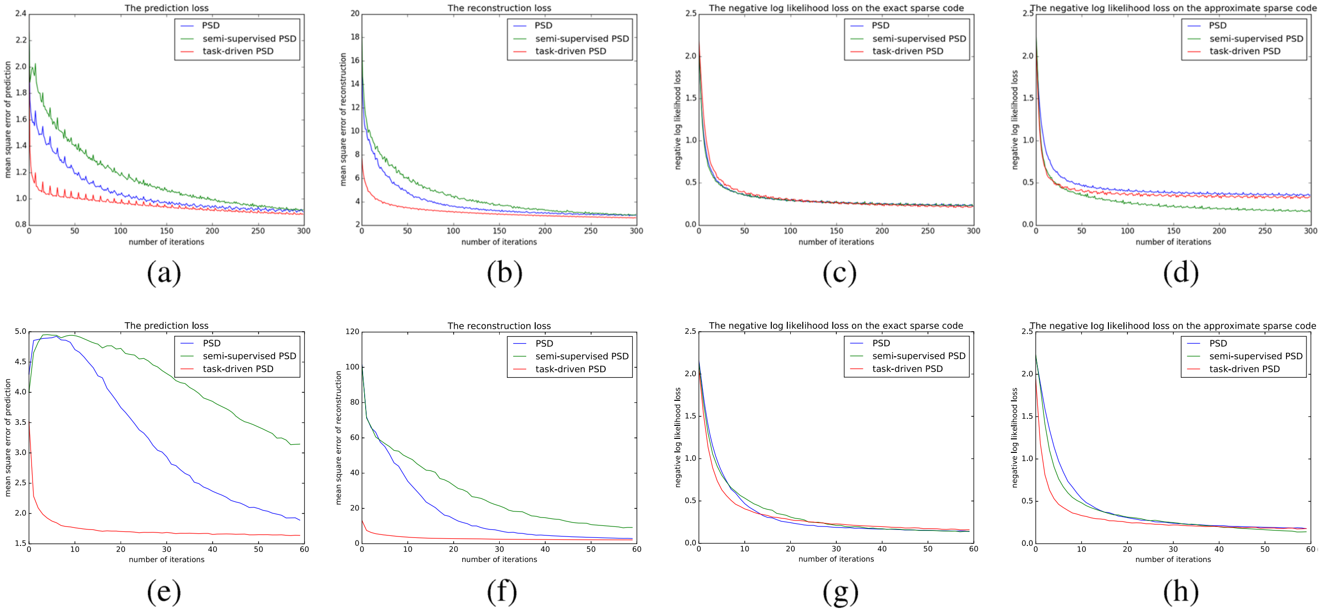


Fig. 1 **a** shows the examples in the MNIST and **b** shows the examples in the USPS

We conduct a series of experiments to compare PSD, semi-supervised PSD, and task-driven PSD. These methods are used to pre-train the single hidden layer neural networks. After the pre-training, the neural networks will be fine-tuned by the back-propagation algorithm.

Firstly, we discuss how to set the hyper-parameters. In [3], Kavukcuoglu et al. have analyzed the effect of λ_2 and suggested to set $\lambda_2 = 1$. This principle is followed in our experiments. We try different λ_1 in $\{0.1, 0.15, 0.2\}$ and the number of hidden units in $\{400, 800, 1200\}$ for all three methods. For the semi-supervised PSD and task-driven PSD, μ controls the tradeoff between the unsupervised and supervised tasks. We set $\mu = 0.5$ to weight them equally. In many dictionary learning methods, the dictionary matrix is initialized by randomly sampled training signals. We also use this trick in our experiments. In addition, we initialize G to be the identity matrix and W, b to be zeros. This way can be seen as a warm-starting from the elastic net. For the pre-training and fine-tuning phases, ρ, a, and r are set to 0.01, 0.0001, and 0.75. The number of iterations is determined by the early stopping strategy. We adopt the batch mode to update the parameters. The batch size is set to 64. The training examples are used for the pre-training and fine-tuning of neural networks. The best hyper-parameters are chosen by the fine-tuned neural networks' recognition performance on the validation set. After the determination of hyper-parameters, we train these models on the whole training set until the convergence of training process. Finally, for both datasets, we choose 800 hidden units and set $\lambda_1 = 0.15$ for all of methods.

Next, we analyze the properties of these methods from different views. $\|x - D\alpha\|_2^2$ is used to evaluate the reconstruction performance of the trained dictionary and the sparse representations. $\|f(x, P_f) - \alpha\|_2^2$ is used to evaluate the prediction accuracy of the regressor. For each method, we train two soft-max classifiers on the exact and approximate sparse representations to predict the labels. The negative log likelihood losses of soft-max classifiers are used to evaluate the discriminative power of representations. We plot the learning curves for these performance indicators in Fig. 2. In order to show the learning process more clearly, we smooth the learning curve by averaging the loss values in



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 2 The training procedure of PSD, semi-supervised PSD, and task-driven PSD. **a**, **b**, **c**, **d** are the learning curve on the MNIST dataset. **e**, **f**, **g**, **h** are the learning curve on the USPS dataset

100 iterations. We also compare the test classification error of the neural networks in Table 1. At last, for the MNIST dataset, we visualize some basis vectors of dictionary and filter matrices in Fig. 3. The elements of dictionary matrix D are shown in the first line and the filters W are in the second line. We can see all of these methods have intuitively explainable features. In [24], Swersky et al. drew an empirical conclusion that nice looking filters will usually correlate with good performance.

PSD PSD is the basis of other algorithms. In [3], Kavukcuoglu illustrates that the PSD can not provide a good reconstruction, but its sparse representations can achieve a high recognition accuracy. The learning process of PSD is plotted by the blue curve in Fig. 2. The experiments on the MNIST dataset show that the PSD has a moderate performance. We can see the prediction and reconstruction errors are small in Fig. 2 a, b. But the discriminative power of internal representations is the worst in Fig. 2c, d. This is because other methods introduce the discriminative information. However, we can see the difference on recognition performance is not obvious. The pre-trained neural network achieve 8.64 % error rate on the test dataset. After the fine-tuning, the error rate is decreased to 2.04 %. For the USPS dataset, the PSD does not work well. Although the performance on the training set is similar to the task-driven PSD, its test error is the worst. We think the reason is that the amount of data in USPS is too small. In this situation, semi-supervised methods usually introduce more information and improve the generalization ability of the representations.

Table 1 The test error rate of three models

Dataset	Without fine-tuning			With fine-tuning		
	PSD	SsPSD	TdPSD	PSD	SsPSD	TdPSD
MNIST	8.64 %	4.90 %	8.14 %	2.04 %	2.31 %	1.98 %
USPS	8.76 %	7.12 %	8.42 %	6.72 %	5.98 %	5.92 %

Semi-supervised PSD The learning curve of semi-supervised PSD is plotted by the green line. Due to the

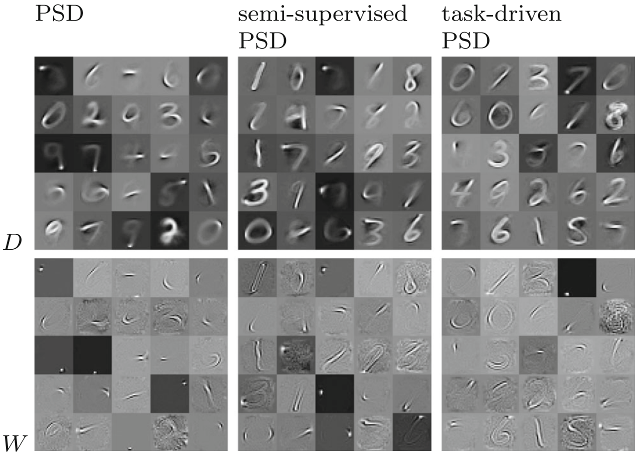


Fig. 3 The visualized dictionaries and filters which is trained on the MNIST dataset

introduction of classification task, the features extracted by the encoder function have the most discriminative power. The negative log likelihood losses in Fig. 2 d, h are obviously lower than other methods. And the pre-trained neural network also achieve the best test error rate. After the fine-tuning, the error rates on two datasets are 2.31 and 5.98 %, respectively. Due to the introduction of classification task, the reconstruction and prediction learning is slow. For the MNIST dataset, the semi-supervised PSD does not enhance the generalization ability of neural network. When the amount of data is small, i.e., the situation of the USPS dataset, the semi-supervised PSD achieves comparable performance to our task-driven PSD.

Task-Driven PSD For our task-driven PSD, the discriminative information is introduced from the optimal sparse representations. Hence, the classification loss on the optimal sparse representations is better than other methods. It is shown in Fig. 2 c, g. The red line is used to plot this learning process. To our surprise, the task-driven PSD achieves the minimal reconstruction loss and prediction loss on both datasets. In other words, the neural network can predict the informative sparse codes very precisely. Additionally, it has the fastest convergence rate. Although our task-driven PSD only achieve a little better performance than other methods, our method can accelerate the learning of dictionary matrix and encoder function. The recognition accuracy on the test dataset is shown in Table 1. The fine-tuned neural network has the lowest error rates.

Discussion

In this paper, we extend the existing work in [3, 9]. The task-driven dictionary learning provides a solid theoretical foundation. However, it is inefficient. Its training involves the Cholesky decomposition. For high dimensional data, the computational cost will be unacceptable. Our experiments are run on a PC with Intel i5-2400 at 3.1 GHz CPU. The training procedures on MNIST and USPS datasets cost 13 and 6 h, respectively. Additionally, the task-driven PSD lacks invariance to small translations in the pixel domain.

For regular-scale datasets and practical applications, many recent developments should be studied in our future work. In [25], Liu et al. propose the multiview Hessian discriminative sparse coding (mHDSC). The label information is treated as an additional view of feature. Hence, the introduction of discriminative information does not increase the computing complexity. Combining mHDSC with PSD would be more efficient than the task-driven PSD. In [26], a scalable optimization algorithm is adopted to make full use of parallel architectures. In many research works, low-level features are input into sparse coding methods rather than the raw pixels. Hence, as proposed in [22], our model can also benefit from convolution. A convolution and pooling layer followed by fully connected linear units can be used as a feature extractor. Then, the task-driven PSD can be applied to these low dimensional features. According to [27, 28], the scope of our work can also be expanded to more complex situations.

Conclusion

In this work, we propose the task-driven PSD to train neural network. Compared with the semi-supervised PSD, our method provides discriminative and representative sparse codes to guide the neural network. After fine-tuning, the neural network can achieve higher recognition accuracy. In the experiments, the reconstruction loss evaluates the representative power of sparse codes. The discriminative power is evaluated by the soft-max loss. The prediction loss indicates whether the neural network can learn the informative sparse codes accurately. The experiment results on MNIST and USPS datasets verify the effectiveness of our method. Finally, the task-driven PSD is a basic extension to existing methods. It is not efficient enough and lacks shift invariance. In the future work, some latest developments will be studied to enhance the practicability of our method.

Acknowledgments This study was funded by the National Natural Science Foundation of China (NSFC) under Grants No. 61273136, No. 61573353 and No. 61533017.

Compliance with Ethical Standards

Conflict of Interest The authors declare that they have no conflict of interest.

Ethical Approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

1. Wei H, Dong Z. V4 neural network model for shape-based feature extraction and object discrimination. Cogn Comput. 2015:1–10.
2. Gros C. Cognitive computation with autonomously active neural networks: An emerging field. Cogn Comput. 2009;1(1):77–90.
3. Kavukcuoglu K, Marc'Aurelio R, LeCun Y. Fast inference in sparse coding algorithms with applications to object recognition. CoRR, abs/1010.3467. 2010.
4. Kavukcuoglu K, Ranzato MA, Fergus R, LeCun Y. Learning invariant features through topographic filter maps. 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 Jun 2009, Miami, Florida, USA; 2009. pp. 1605–1612.
5. Kavukcuoglu K, Sermanet P, Boureau Y-L, Gregor K, Mathieu M, LeCun Y. Learning convolutional feature hierarchies for visual recognition. Advances in Neural Information Processing

Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 Dec 2010, Vancouver, British Columbia, Canada; 2010. pp. 1090–1098.

6. He B, Xu D, Nian R, Heeswijk M, Yu Q, Yoan M, Amaury L. Fast face recognition via sparse coding and extreme learning machine. Cogn Comput. 2014;6(2):264–277.
7. Bengio Y, Courville AC, Pascal V. Unsupervised feature learning and deep learning: a review and new perspectives. CoRR, abs/1206.5538. 2012.
8. Bengio Y, Lamblin P, Popovici D, Larochelle H. Greedy layer-wise training of deep networks. Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia Canada, Dec 4-7 2006; 2006. pp. 153–160.
9. Mairal J, Bach FR, dictionary JP. Task-driven learning. IEEE Trans Pattern Anal Mach Intell. 2012;34(4):791–804.
10. Mallat S, Zhang Z. Matching pursuits with time-frequency dictionaries. IEEE Trans Signal Process. 1993;41(12):3397–3415.
11. Tibshirani R. Regression shrinkage and selection via the lasso. J R Stat Soc. 1994;58(1):267–288.
12. Chen SS, Donoho DL, Saunders MA. Atomic decomposition by basis pursuit. SIAM Rev. 2001;43(1):129–159.
13. Zou H, Hastie T. Regularization and variable selection via the elastic net. J R Stat Soc. 2005:67.
14. Shalev-Shwartz S, Tewari A. Stochastic methods for l_1-regularized loss minimization. J Mach Learn Res. 2011;12:1865–1892.
15. Efron B, Tibshirani R. Least angle regression. Ann Stat. 2004;32(2):2004.
16. Mallat S. A wavelet tour of signal processing (2.ed.). Academic Press. 1999.
17. Starck J-L, Candès EJ, Donoho DL. The curvelet transform for image denoising. IEEE Trans Image Process. 2002;11(6):670–684.
18. Do MN, Vetterli M. The contourlet transform: an efficient directional multiresolution image representation. IEEE Trans Image Process. 2005;14(12):2091–2106.
19. Aharon M, Elad M, Brucstein A. k –svd: An algorithm for designing overcomplete dictionaries for sparse representation. IEEE Trans Signal Process. 2006;54(11):4311–4322.
20. Olshausen BA, Field DJ. Sparse coding with an overcomplete basis set: A strategy employed by v1?. Vis Res. 1997;37(23):3311–3325.
21. Lee H, Battle A, Raina R, Ng AY. Efficient sparse coding algorithms. Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia Canada, Dec 4-7 2006; 2006. pp. 801–808.
22. Ba J, Caruana R. Do deep nets really need to be deep? Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada; 2014. pp. 2654–2662.
23. Fuchs J-J. Recovery of exact sparse representations in the presence of bounded noise. IEEE Trans Inf Theory. 2005;51(10):3601–3608.
24. Swersky K, Ranzato MA, Buchman D, Marlin BM, Freitas N. On autoencoders and score matching for energy based models. Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, Jun 28 - Jul 2, 2011; 2011. pp. 1201–1208.
25. Liu W, Tao D, Cheng J, Tang Y. Multiview hessian discriminative sparse coding for image annotation. Comput Vis Image Underst. 2014;118:50–60.
26. Xum C, Tao D, Xu C. Robust extreme multi-label learning. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA August 13-17, 2016; 2016. pp. 1275–1284.
27. Liu W, Zha Z-J, Wang Y, Lu K, Tao D. P-laplacian regularized sparse coding for human activity recognition. IEEE Trans Ind Electron. 2016;63(8):5120–5129.
28. Qiao M, Liu L, Yu J, Xu C, Tao D. Diversified dictionaries for multi-instance learning. Pattern Recognition. 2016.