Submodular Dictionary Learning for Sparse Coding

Zhuolin Jiang[†], Guangxiao Zhang[†§], Larry S. Davis[†]
[†]Institute for Advanced Computer Studies, University of Maryland, College Park, MD, 20742
[§]Global Land Cover Facility, University of Maryland, College Park, MD, 20742
{zhuolin, gxzhang, lsd}@umiacs.umd.edu

Abstract

A greedy-based approach to learn a compact and discriminative dictionary for sparse representation is presented. We propose an objective function consisting of two components: entropy rate of a random walk on a graph and a discriminative term. Dictionary learning is achieved by finding a graph topology which maximizes the objective function. By exploiting the monotonicity and submodularity properties of the objective function and the matroid constraint, we present a highly efficient greedy-based optimization algorithm. It is more than an order of magnitude faster than several recently proposed dictionary learning approaches. Moreover, the greedy algorithm gives a near-optimal solution with a $(1/2)$-approximation bound. Our approach yields dictionaries having the property that feature points from the same class have very similar sparse codes. Experimental results demonstrate that our approach outperforms several recently proposed dictionary learning techniques for face, action and object category recognition.

1. Introduction

Sparse coding represents an input signal y as a linear combination of a few items from a dictionary, D. Learning the dictionary is critical for good performance. The SRC algorithm [30], for example, employs the entire set of training samples as the dictionary and achieves impressive performance on face recognition. For many applications, however, rather than using the entire set of training data as dictionary, it is computationally advantageous to learn a compact dictionary from training data.

Many algorithms [30, 1, 35, 12, 5] have been developed for learning such dictionaries. The dictionary in [30] is manually selected from the training samples. [1] proposes the K-SVD algorithm using orthogonal matching pursuit for sparse coding and learns an over-complete dictionary in a reconstructive manner. The method of optimal directions (MOD) [5] shares the same sparse coding step but has a different dictionary update process. These learning approaches have led to excellent results in image denoising and inpainting [22, 1]. However, the dictionaries obtained by using the reconstructive formulation, while optimized for reconstruction, may not necessarily be best for classification. The reconstructive formulation can be augmented with discriminative terms [12, 35, 32, 21, 25], that encourage learning dictionaries relevant to classification.

Generally, most dictionary learning algorithms [12, 35, 32, 21, 25, 27] iteratively alternate a sparse coding step (for computing the sparse codes over the dictionary obtained at the previous iteration), with a dictionary update step given the sparse codes from the previous sparse coding step. They access the entire training set at each iteration in order to optimize the objective function. These algorithms converge slowly when the reconstruction error is required to be small. Moreover, they may get trapped in local minimum. Although some algorithms [16, 22, 29] for efficient dictionary learning have been proposed recently, effectively solving the optimization problem for dictionary learning is still a significant computational challenge.

Submodularity can be considered as a discrete analog of convexity. The diminishing return property of submodularity has been employed in applications such as sensor placement [10], superpixel segmentation [19] and clustering [36, 23]. In this paper, we present a supervised algorithm for efficiently learning a compact and discriminative dictionary for sparse representation. We define a novel monotonic and submodular objective function, which consists of two terms: the entropy rate of a random walk on a graph and a discriminative term. The entropy rate term favors compact and homogeneous clusters, so each cluster center can well represent the cluster elements. The discriminative term encourages the clusters to be class pure and leads to a smaller number of clusters. Hence, each cluster center preserves category information and makes the computed sparse codes discriminative when using a dictionary consisting of all the cluster centers. As shown in Figures 1(b) and 2(b), the dictionaries learned by our approach encourage the sparse codes for signals from the same class to be similar. The main contributions of this paper are:

- We model the problem of discriminative dictionary learning as a graph topology selection problem, which is solved by maximizing a monotonically increasing and submodular objective function.
- We present a highly efficient greedy optimization approach by using the submodularity and monotonic increasing properties of the objective function under a matroid constraint, that scales to large datasets.
- We prove that the objective function is monotonically increasing and submodular, and the cycle-free con-

straint and the connected component constraint in the graph partitioning induce a matroid. The proofs are given in the supplementary material.

- Our approach achieves state of the art performance on various public face, action and object recognition benchmarks.

1.1. Related Work

Discriminative dictionary learning algorithms have been the focus of recent research. Some algorithms learn multiple dictionaries or category-dependent dictionaries [20, 33]. [20] constructs one dictionary per class and classification is based on the corresponding reconstruction errors.

Some algorithms learn a dictionary by merging or selecting dictionary items from a large set of dictionary item candidates [18, 14, 26, 13]. [18, 14] learn a dictionary through merging two items by maximizing the mutual information of class distributions. [13] constructs a dictionary for signal reconstruction from a set of dictionary item candidates. The objective function in [13] satisfies approximate submodularity under a certain incoherence assumption on the candidate elements. For these approaches, a large dictionary is required at the beginning to guarantee discriminative power of the constructed compact dictionary.

Some algorithms add discriminative terms into the objective function of dictionary learning [27, 35, 12, 32, 21, 25]. The discriminative terms include Fisher discrimination criterion [27], linear predictive classification error [35, 25], optimal sparse coding error [12], softmax discriminative cost function [21] and hinge loss function [32].

In addition, submodularity has been exploited for clustering [36, 23]. [23] uses single linkage and minimum description length criteria to find an approximately optimal clustering. [36] analyzes greedy splitting for submodular clustering. Both of them solve a submodular function minimization problem in polynomial time to learn dictionaries.

Compared to these approaches, our approach maximizes a monotonically increasing and submodular objective function to learn a single compact and discriminative dictionary. The solution to the objective function is efficiently achieved by simply employing a greedy algorithm. Our approach adaptively allocates different numbers of dictionary items to each class so it has good representational power to represent signals with large intra-class difference, in contrast to those approaches such as [12] which uniformly allocate dictionary items to each class. Moreover, unlike algorithms such as [1, 35, 12], our approach does not require a good initial solution, and the computed solution is guaranteed to be a $(1/2)$-approximation to the optimal solution [24].

2. Sparse Coding and Dictionary Learning

Let Y be a set of N input signals from a n-dimensional feature space, *i.e.* $Y = [y_1...y_N] \in R^{n \times N}$. Assuming a dictionary D of size K is given, the sparse representations $Z = [z_1...z_N] \in R^{K \times N}$ for Y are obtained by solving:

$$Z = \arg\min_{Z} \|Y - DZ\|_2^2 \quad s.t.\ \forall i,\ \|z_i\|_0 \leq s \quad (1)$$

where $\|Y - DZ\|_2^2$ denotes the reconstruction error and $\|z_i\|_0 \leq s$ is the sparsity constraint (that each signal has s or fewer items in its decomposition). Orthogonal matching pursuit algorithm can be used to solve (1).

The performance of sparse representation depends critically on D. SRC [30] constructs D by using all the training samples; this can result in a very large dictionary, which makes subsequent sparse coding expensive. Thus, methods for learning a small-size dictionary for sparse coding have been proposed. For example, the K-SVD algorithm [1] is well-known for efficiently learning an over-complete dictionary from a set of training signals; it solves:

$$< D, Z >= \arg\min_{D,Z} \|Y - DZ\|_2^2 \quad s.t.\ \forall i,\ \|z_i\|_0 \leq s \quad (2)$$

where $D = [d_1...d_K] \in R^{n \times K}$ is the learned dictionary, and Z are the sparse representations of Y.

However, K-SVD only focuses on minimizing the reconstruction error and does not consider the dictionary's utility for discrimination. The discrimination power should be considered when constructing dictionaries for classification tasks. Some discriminative approaches [25, 35, 12, 21, 32], which add extra discriminative terms into the cost function for classification task, have been investigated.

However, these approaches appear to require large dictionaries to achieve good performance, leading to high computation cost especially when additional terms are added into the objective function. The computation requirements are further aggravated when good classification results are based on multiple pairwise classifiers such as [21, 32].

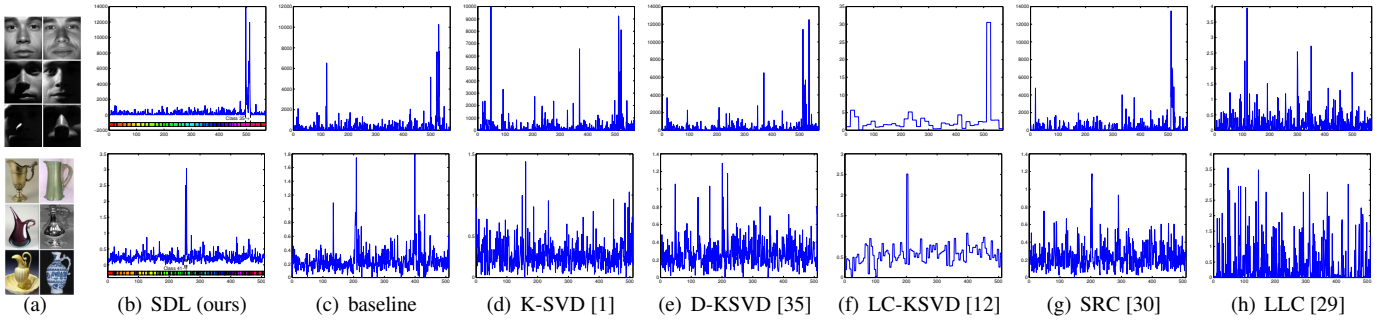
We will show that good classification results can be obtained using only a compact, single unified dictionary, which is trained in a highly efficient way. The sparse representation for classification assumes that a new test sample can be well represented by the training samples from the same class [30]. If the class distributions for each dictionary item are highly peaked in one class, then this implicitly generates a label for each dictionary item. This leads to a discriminative sparse representation over the learned dictionary; additionally, the learned dictionary has good representational power because the dictionary items are the cluster centers (*i.e.*, spanning all the subspaces of object classes).

3. Submodular Dictionary Learning

We consider dictionary learning as a graph partitioning problem, where data points and their pairwise relationships are mapped into a graph. To partition a graph into K clusters, we search for a graph topology that has K connected components and maximizes the objective function.

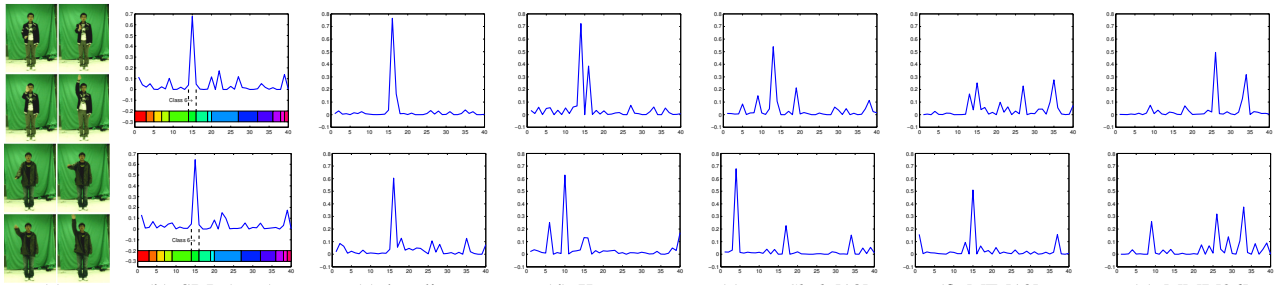
3.1. Preliminaries

Submodularity: [24] Let E be a finite set. A set function $F : 2^E \rightarrow R$ is submodular if $F(A \cup \{a_1\}) - F(A) \geq$



(a) (b) SDL (ours) (c) baseline (d) K-SVD [1] (e) D-KSVD [35] (f) LC-KSVD [12] (g) SRC [30] (h) LLC [29]

Figure 1. Examples of sparse codes using dictionaries learned by different approaches on the Extended YaleB and Caltech101 datasets. Each waveform indicates the sum of absolute sparse codes for different testing samples from the same class. The first and second row correspond to class 35 in Extended YaleB (32 testing frames) and class 41 in Caltech101 (55 testing frames) respectively. (a) are sample images from these classes. Each color from the color bar in (b) represents one dominating class of the class distributions for dictionary items. Different from [12] which uniformly constructs dictionary items for each class, our approach adaptively allocates different numbers of dictionary items to each class. The black dashed lines demonstrate that the curves are highly peaked in one class. The figure is best viewed in color and 600% zoom in.



(a) (b) SDL (ours) (c) baseline (d) K-means (e) *Liu-Shah* [18] (f) ME [10] (g) MMI [26]

Figure 2. Examples of sparse codes using dictionaries learned by different approaches for two action sequences from the Keck gesture dataset. Each waveform indicates the average of absolute sparse codes for each action sequence. The first and second row correspond to gesture 6 from person 2 and person 3 respectively. (a) are sample images from these sequences. Each color from the color bar in (b) represents one dominating class of the class distributions for dictionary items. The black dashed lines demonstrate that the curves are highly peaked in one class. The figure is best viewed in color and 600% zoom in.

$F(A \cup \{a_1, a_2\}) - F(A \cup \{a_2\})$, for all $A \subseteq E$ and $a_1, a_2 \in E \backslash A$. This property is referred to as diminishing returns, stating that adding an element to a smaller set helps more than adding it to a larger set.

Matroid: [3] Let E be a finite set and $\mathcal{I}$ a collection of subsets of E. A matroid is an ordered pair $\mathcal{M} = (E, \mathcal{I})$ satisfying three conditions: (a) $\emptyset \in \mathcal{I}$; (b) If $A \subseteq B$ and $B \in \mathcal{I}$, then $A \in \mathcal{I}$; (c) If $A \in \mathcal{I}$, $B \in \mathcal{I}$ and $|A| < |B|$, then there is an element $x \in B - A$ such that $A \cup x \in \mathcal{I}$.

3.2. Graph Construction

We map a dataset Y into an undirected k-nearest-neighbor graph $G = (V, E)$, where vertex set V denotes the data points, and edge set E models the pairwise relationships between data points. Let v_i be the i-th vertex and $e_{i,j}$ be an edge that connects v_i and v_j. The similarity between vertices is given by a weight function $w : E \rightarrow R^+ \cup \{0\}$. The edge weighs are symmetric for an undirected graph, *i.e.* $w_{i,j} = w_{j,i}$. We set $w_{i,j} = 0$ if there is no edge joining v_i and v_j. We use a Gaussian kernel to convert pairwise distances to similarities: $w_{i,j} = \exp(-\beta d^2(v_i, v_j))$, where $d(v_i, v_j)$ is the distance between v_i and v_j. The normalization factor β is set to $\beta = (2\langle d^2(v_i, v_j)\rangle)^{-1}$, where $\langle\cdot\rangle$ denotes expectation over all pairwise distances, as in [28].

Our aim is to select a subset A of E ($A \subseteq E$) resulting in graph $G = (V, A)$ consisting of K connected components. In addition, each vertex v_i of G has a self-loop $e_{i,i}$. These self-loops are necessary for the proposed random walk model, although they do not affect graph partitioning. When an edge $e_{i,j}$ is not included in A, its weight $w_{i,j}$ is redistributed to the self-loops of v_i and v_j. The edge weights $w_{i,i}$ and $w_{j,j}$ of the self-loops are then given by: $w_{i,i} = w_{i,i} + w_{i,j}$, $w_{j,j} = w_{j,j} + w_{i,j}$. This keeps the total incident weight for each vertex constant, so that the distribution $\boldsymbol{\mu}$ in (3), below, for the graph is stationary.

3.3. Entropy Rate of A Random Walk

We use the entropy rate of the random walk to obtain compact and homogeneous clusters so that the constructed dictionary has good representative power. The entropy rate measures the uncertainty of a stochastic process $\mathbf{X} = \{X_t | t \in T\}$ where T is an index set. It can be defined as: $\mathcal{H}(\boldsymbol{X}) = \lim_{t\rightarrow\infty} H(X_t|X_{t-1}...X_1)$, which is the conditional entropy of the last random variable given the past. For a stationary 1st-order Markov chain, the entropy rate is given by: $\mathcal{H}(\mathbf{X}) = \lim_{t\rightarrow\infty} H(X_t|X_{t-1}) =$

$\lim_{t\to\infty} H(X_2|X_1) = H(X_2|X_1)$.

A random walk $\mathbf{X} = \{X_t|t \in T, X_t \in V\}$ is a sequence of vertices of the graph. We use the random walk model from [4] with a transition probability from v_i to v_j: $P_{i,j} = w_{i,j}/w_i$, where $w_i = \sum_{j:e_{i,j}\in E} w_{i,j}$ is the total incident weight of v_i, and the stationary distribution is:

$$\boldsymbol{\mu} = (\mu_1, \mu_2, ..., \mu_{|V|})^t = (\frac{w_1}{w_{all}}, \frac{w_2}{w_{all}}, ..., \frac{w_{|V|}}{w_{all}})^t \quad (3)$$

where $w_{all} = \sum_{i=1}^{|V|} w_i$ is the sum of the total weights of all vertices. The entropy rate of the random walk in [4] is defined as: $\mathcal{H}(\boldsymbol{X}) = H(X_2|X_1) = \sum_i \mu_i H(X_2|X_1 = v_i) = -\sum_i \mu_i \sum_j P_{i,j} \log P_{i,j}$. It is a sum of the entropies of the transition probabilities weighted by μ_i.

The proposed graph construction leaves $\boldsymbol{\mu}$ in (3) unchanged. The set functions for the transition probabilities $P_{i,j} : 2^E \to R$ with respect to A are defined as:

$$P_{i,j}(A) = \begin{cases} 1 - \frac{\sum_{j:e_{i,j}\in A} w_{i,j}}{w_i} & \text{if } i = j, \\ \frac{w_{i,j}}{w_i} & \text{if } i \neq j,\ e_{i,j} \in A, \\ 0 & \text{if } i \neq j,\ e_{i,j} \notin A. \end{cases} \quad (4)$$

Consequently, we can define a set function for the entropy rate of the random walk on $G = (V, A)$ as:

$$\mathcal{H}(A) = -\sum_i \mu_i \sum_j P_{i,j}(A) \log(P_{i,j}(A)). \quad (5)$$

Intuitively, given μ_i, maximizing the entropy rate in (5) encourages the edges from each v_i to have similar weights, *i.e.*, similar transition probabilities to its connected vertices. Similarly, given the entropies of transition probabilities, maximizing the entropy rate in (5) encourages the edges from v_i which have large weight (small distance) to be selected. Hence the entropy rate of the random walk on the graph can generate compact and homogeneous clusters, as shown in Figure 3. It results in dictionary items that represent the input signals well.

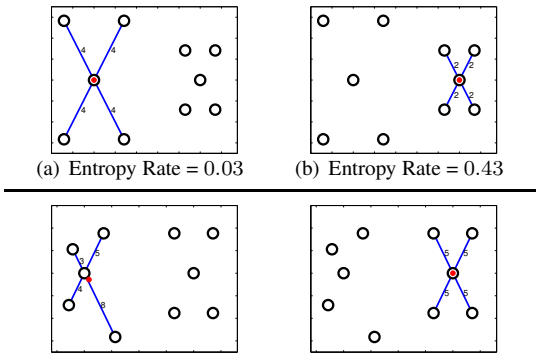
The entropy rate of a random walk $\mathcal{H} : 2^E \to R$ on the proposed constructed graph is a monotonically increasing and submodular function. Monotonicity is obvious because the addition of any edge to A increases the uncertainty of a jump in a random walk. Submodularity is related to the observations that the increase of the uncertainty from selecting an edge $e_{i,j}$ is less in a later stage because it will be shared with more edges connected to v_i or v_j [19]. The proof is given in the supplementary material based on [19].

3.4. Discriminative Function

We construct a discriminative function that encourages the sparse representations of signals over learned dictionary (*i.e.* cluster centers), from the same class to be similar.

Let m denote the number of object categories. $\mathcal{N} = [\mathbf{N}^1...\mathbf{N}^N] \in R^{m\times N}$ is a count matrix for the number of elements from each object class assigned to each cluster [1]. Let $\mathbf{N}^i = [N_1^i...N_m^i]^t$, where N_m^i is the number of objects from the m-th class that were assigned to the i-th cluster.

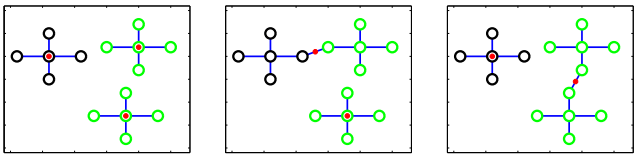
[1] each input signal can be considered as an individual cluster here.



(a) Entropy Rate = 0.03 (b) Entropy Rate = 0.43

(c) Entropy Rate = 0.22 (d) Entropy Rate = 0.24

Figure 3. The importance of the entropy rate for constructing a dictionary with good representative power. The number next to the edges is a distance between vertices. Each vertex has a self loop which is not shown. Each figure outputs six clusters shown as connected components. The red dots are cluster centers. We do not show the red dots for those clusters which only contain one element. By computing the entropy rate, we observe that the more compact clusters in (b) are preferred compared to (a), while the more symmetric cluster, that is, homogeneous cluster in (d) is preferred compared to (c). It demonstrates that the cluster centers, *i.e.* dictionary items, in (b)(d) have good representative power.



(a) Disc. Fun. = -2.00 (b) Disc. Fun. = -1.33 (c) Disc. Fun. = -1.00

Figure 4. The importance of the discriminative term for constructing a dictionary with discriminative capabilities. The colors of empty circles denote different categories of data points. The connected components show different clusters. The red dots in the figure are cluster centers. The green and black points in (b) are grouped into the same word and consequently cannot be distinguished. The green points in (a) are separated into two words and as a result the sparse representations for other green points might not be similar. The more 'class pure' clustering in (c) has a higher objective value than (b). The fewer number of clusters in (c) has a higher objective value than (a). Sparse codes obtained by the cluster centers in (c) are best for classification. Hence the discriminative term suggests the higher discriminative power for the cluster centers in (c).

Let N_A denote the number of connected components or subgraphs. The graph partitioning for A is $\mathcal{S}_A = \{S_1...S_{N_A}\}$. The class purity for cluster S_i is computed as: $\mathcal{P}(S_i) = \frac{1}{C_i} \max_y N_y^i$, where y denotes the class label, $y \in \{1...m\}$, and $C_i = \sum_y N_y^i$ is the total count for objects of all classes assigned to cluster i. Thus the overall purity for $\mathcal{S}_A$ is:$\mathcal{P}(\mathcal{S}_A) = \sum_{i=1}^{N_A} \frac{C_i}{C} \mathcal{P}(S_i) = \sum_{i=1}^{N_A} \frac{1}{C} \max_y N_y^i$, where $C = \sum_i C_i$. The discriminative term is defined as:

$$\mathcal{Q}(A) \equiv \mathcal{P}(\mathcal{S}_A) - N_A = \sum_{i=1}^{N_A} \frac{1}{C} \max_y N_y^i - N_A. \quad (6)$$

The purity term $\mathcal{P}(\mathcal{S}_A)$ encourages clusters with pure

class labels, while N_A favors a smaller number of clusters. The class pure cluster encourages similar feature points to be represented by the same cluster centers (*i.e.*, dictionary items). Hence, it should result in high discrimination power for the dictionary items. From Figure 4, the more class pure cluster is preferred for a fixed number of clusters, while a fewer number of clusters is preferred for a fixed purity. Similar to the entropy rate, the discriminative term $\mathcal{Q} : 2^E \rightarrow R$, is also a monotonically increasing and submodular function.

The final objective function combines the entropy rate and the discriminative term for dictionary learning, *i.e.*, $\mathcal{F}(A) \equiv \mathcal{H}(A) + \lambda\mathcal{Q}(A)$. The solution is achieved by maximizing the objective function with A selected as:

$$\max_A \mathcal{H}(A) + \lambda\mathcal{Q}(A) \; s.t. \; A \subseteq E \; and \; N_A \geq K, \qquad (7)$$

where λ controls the relative contribution between entropy rate and the discriminative term. $N_A \geq K$ is a constraint on the number of connected components. This constraint enforces exactly K connected components because the objective function is monotonically increasing. λ is given by $\lambda = \gamma\lambda' = \left(\frac{\max_{e_{i,j}} \mathcal{H}(e_{i,j}) - \mathcal{H}(\emptyset)}{\max_{e_{i,j}} \mathcal{Q}(e_{i,j}) - \mathcal{Q}(\emptyset)}\right)\lambda'$, where λ' is a predefined parameter and γ is the ratio of maximal entropy rate increase and the maximal discriminative term increase when adding a single edge to the graph.

The objective function in (7) is submodular and monotonically increasing, because it is a linear combination of monotonic and submodular functions [24]. This makes the clusters more compact and class pure. Their cluster centers represent the input signals well and preserve object category information. Hence when using these cluster centers as dictionary items, our approach encourages the computed sparse codes for signals from the same class to be similar, as shown in Figures 1(b) and 2(b).

Algorithm 1 Submodular Dictionary Learning (SDL)

Input: $G = (V, E)$, w, K, λ and $\mathcal{N}$
Output: D
Initialization: $A \leftarrow \emptyset$, $D \leftarrow \emptyset$
for $N_A > K$ **do**
  $\tilde{e} = \underset{A \cup \{e\} \in \mathcal{I}}{\operatorname{argmax}} \mathcal{F}(A \cup \{e\}) - \mathcal{F}(A)$
  $A \leftarrow A \cup \{\tilde{e}\}$
end for
for each subgraph S_i in $G = (V, A)$ **do**
  $D \leftarrow D \cup \{\frac{1}{|S_i|}\sum_{j:v_j \in S_i} v_j\}$
end for

3.5. Optimization

Direct maximization of a submodular function is a NP-hard problem. However, one simple approximate solution can be obtained by a greedy algorithm from [24]. The algorithm starts from a empty set $A = \emptyset$ and iteratively adds edges to A. It always chooses the edge which provides the largest gain for $\mathcal{F}$ at each iteration. The iterations stop when the desired number of connected components is obtained.

Note that the edges that induce cycles in the graph have no effect on the graph partition. In order to speed up the optimization, a cycle-free constraint is used here so that those edges are not included in A. This greatly reduces the evaluations in the greedy search and leads to a smaller solution space. The cycle-free constraint and the connected component constraint in the graph partitioning, *i.e.* $N_A \geq K$ induce a matroid $\mathcal{M} = (E, \mathcal{I})$. $\mathcal{I}$ is the collection of subsets $A \subseteq E$ which satisfies: A includes no cycles and the graph partition from A has more than K connected components.

Maximization of a submodular function with a matroid constraint gives a $1/2$-approximation bound on the optimality of the solution [24]. Hence the greedy algorithm can obtain a performance-guaranteed solution. Algorithm 1 presents the pseudocode of our algorithm.

3.6. Algorithm Complexity

The complexity of a naive implementation of Algorithm 1 is $O(|V|^2)$, because it iterates $O(|V|)$ times to select edges, and evaluates the whole edge list to select the edge which has the largest gain at each iteration. Actually we can further exploit the submodularity property of the objective function to speed up the optimization.

We first evaluate the gain in the objective function for each edge and construct a max heap structure. The edge with the largest gain is selected as the top element from the heap at each iteration. The addition of any edge into A impacts the gains of the remaining edges. Here, instead of recomputing the gains for every remaining edge in the heap, we perform lazy evaluation which only updates the gain of the top element. The key idea is to realize that the gain for each edge can never increase because of the diminishing return property. In many cases, the recomputation of gain for the top element is not much smaller, hence the top element will stay the top element even after the update.

The worst case is that we need to update the gain for each edge and then re-establish the heap property after the addition of any edge into A. That is, we need to rebuild the heap, which takes $O(|V| \log |V|)$ time, hence the worse case is $O(|V|^2 \log |V|)$ [3]. In fact, only a few updates are ever performed in the heap at each iteration, hence the complexity of our algorithm is effectively $O(|V| \log |V|)$ [2].

4. Classification Approach

4.1. Face and Object Recognition

Our dictionary learning algorithm forces the signals from the same class to have very similar sparse representations, which results in good classification performance even using a simple linear classifier. For face and object recognition in static images, we employ the multivariate ridge regression

[2]The complexity here does not include the complexity of constructing the k-nearest-neighbor graph G, which is an input to our algorithm.

model with the quadratic loss and L_2 norm regularization:

$$W = \arg\min_{W} ||H - WZ||^2 + \alpha||W||_2^2, \qquad (8)$$

where $W \in R^{m\times K}$ denotes the linear classifier parameters and $H = [h_1...h_N] \in R^{m\times N}$ are the class label matrix of input signals Y. $h_i = [0, 0...1...0, 0]^t \in R^m$ is a label vector corresponding to an input signal y_i, where the non-zero position indicates the class of y_i. (8) yields the solutions: $W = (ZZ^t + \alpha I)^{-1}ZH^t$.

For a test image y_i, we first compute its sparse representation z_i using (1). Then we simply use W in (8) obtained from training data to estimate a class label vector: $l = Wz_i$, where $l \in R^m$. The label of y_i is the index i where l_i is the largest element of l.

4.2. Action Classification

For the classification of action sequences, we first compute the sparse representations for each frame, then employ dynamic time warping (DTW) to align and measure the distance between two sequences in the sparse representation domain; next a K-NN classifier is used for recognition.

5. Experiments

We evaluate our approach on the Extended YaleB database [8], Caltech101 [6] and Keck gesture dataset [17]. We compare our results with K-SVD [1], D-KSVD [35], LC-KSVD [12], SRC [30], and LLC [29] for learning a dictionary directly from the training data on the Extended YaleB and Caltech101 dataset, while we compare our results with K-means, *Liu-Shah* [18], MMI [26] and ME [10] for learning a dictionary from a large set of dictionary item candidates on the Keck gesture dataset. The baseline approach is to replace our discriminative function with a balance function described in [19], which aims to balance the size of each cluster. We refer to it and our approach as 'baseline' and 'SDL', respectively, in the following.

5.1. Dictionary Learning from the Training Data

5.1.1 Extended YaleB Database

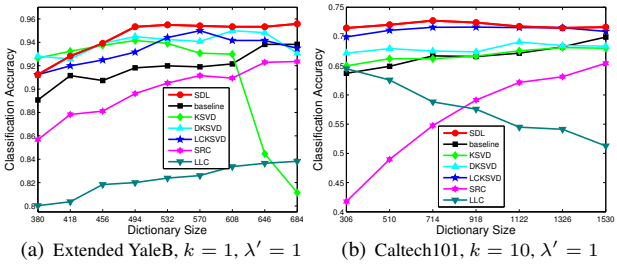
The Extended YaleB database contains $2,414$ frontal face images of 38 persons [8]. There are 64 images with a size of 192×168 pixels for each person. This dataset is challenging due to varying illuminations and expressions. We randomly select half of the images as training and the other half for testing. Following the setting in [35], each face is represented as a 504 sized random-face feature vector.

We evaluate our approach in terms of the classification accuracy using dictionary sizes 380 to 684, and compare it with baseline, K-SVD [1], D-KSVD [35], SRC [3] [30], LLC [29] and the recently proposed LC-KSVD [12]. The sparsity s is set to 30. Figure 5(a) shows that our approach outperforms the state of the art approaches in terms of classification accuracy with different dictionary sizes. The classification accuracy is 95.9% when SRC directly uses all the training data as the dictionary (The result of our approach is 96.4%, which only shows that the linear classifier in (8) used in SDL is slightly better than reconstruction error based classifier used in SRC).

[3]We randomly choose the average of dictionary size per person from each person and report the best performance achieved.

Table 1. Computation time (s) for dictionary training using random-face features on the Extended YaleB database.

Dict. size	418	456	494	532	570	608	646	684
SDL	0.9	1.0	0.9	0.9	0.9	1.0	0.9	0.9
K-SVD [1]	52.6	56.1	59.8	64.9	67.9	72.2	76.2	78.0
D-KSVD [35]	53.1	56.9	60.5	65.8	68.1	74.9	77.6	79.2
LC-KSVD [12]	67.2	72.6	78.3	86.5	90.7	97.8	104.4	112.3



(a) Extended YaleB, $k = 1, \lambda' = 1$ (b) Caltech101, $k = 10, \lambda' = 1$

Figure 5. Classification accuracy performances using different approaches with different dictionary sizes. (a) Performance on the Extended YaleB; (b) Performance on the Caltech101.

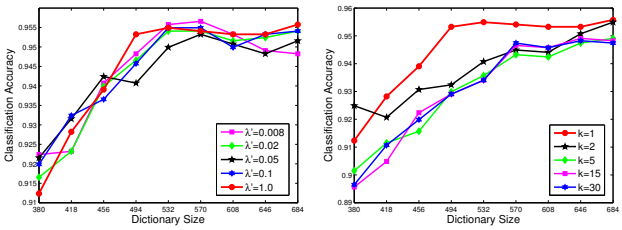
In Figure 6(a), we plot the performance curves for a range of λ' for a fixed $k = 1$ (recall k is the number of nearest neighbors used to construct the original graph). We observe that our approach is insensitive to the selection of λ'. Hence we use $\lambda' = 1.0$ throughout the experiments. Figure 6(b) shows the performance curves for a range of k for $\lambda' = 1.0$. We observe that a smaller value of k results in better performances. This also leads to more efficient dictionary learning since the size of the heap is proportional to the size of the edge set of G.

We also compare the computation time of training a dictionary with K-SVD, D-KSVD and LC-KSVD. The parameter k is set to 1 since we get good performance in Figure 6(b). In Table 1, our approach is at least 50 times faster than the state of art dictionary learning approaches.

5.1.2 Caltech101 Dataset

The Caltech101 dataset [6] contains 9144 images from 102 classes (*i.e.* 101 object classes and a 'background' class). The samples from each class have significant shape differences. Following [12], we extract spatial pyramid features [15] for each image and then reduce them to 3000 dimensions by PCA. The sparse codes are computed from spatial pyramid features. Following the common experimental protocol, we train on randomly selected 5, 10, 15, 20, 25 and 30 samples per category and test on the rest. We repeat the experiments 10 times and the final results are reported as the average and standard deviation of each run.

We evaluate our approach and compare the results with state-of-art approaches [30, 1, 35, 12, 34, 15, 9, 2, 11, 25, 7, 31, 29]. The sparsity s is set to 30. The comparative



(a) Varying λ' for $k = 1$ (b) Varying k for $\lambda' = 1$

Figure 6. Effects of parameter selection of λ' and k on the classification accuracy performance on the Extended YaleB database.

Table 2. Classification accuracies using spatial pyramid features on the Caltech101. SDL also provides the standard deviations.

Training Images	5	10	15	20	25	30
Malik [34]	46.6	55.8	59.1	62.0	-	66.20
Lazebnik [15]	-	-	56.4	-	-	64.6
Griffin [9]	44.2	54.5	59.0	63.3	65.8	67.60
Irani [2]	-	-	65.0	-	-	70.40
Grauman [11]	-	-	61.0	-	-	69.10
Venkatesh [25]	-	-	42.0	-	-	-
Gemert [7]	-	-	-	-	-	64.16
Yang [31]	-	-	67.0	-	-	73.20
Wang [29]	51.15	59.77	65.43	67.74	70.16	73.44
SRC [30]	48.8	60.1	64.9	67.7	69.2	70.7
K-SVD [1]	49.8	59.8	65.2	68.7	71.0	73.2
D-KSVD [35]	49.6	59.5	65.1	68.6	71.1	73.0
LC-KSVD [12]	54.0	63.1	**67.7**	70.5	72.3	73.6
SDL	**55.3** $\pm$0.5	**63.4** $\pm$ 0.5	67.5 $\pm$ 0.3	**70.7** $\pm$ 0.3	**73.1** $\pm$ 0.4	**75.3** $\pm$ 0.4

results are shown in Table 2. Our approach outperforms all the competing approaches except the comparison with LC-KSVD in the case of 15 training samples per category.

We randomly select 30 images per category as training images, and evaluate our approach with different dictionary sizes from 306 to 1530. We also compare our results with the baseline, K-SVD [1], D-KSVD [35], LC-KSVD [12], SRC [30] and LLC [29]. Figure 5(b) shows that our approach outperforms the competing K-SVD, D-KSVD, SRC, LLC and is comparable to LC-KSVD.

We also compare the computation time of training a dictionary with the K-SVD, D-KSVD and LC-KSVD. The parameter k (for k-nearest-neighbor graph construction) is 10. As shown in Table 3, our approach can train approximately 15 times faster when we construct a dictionary of size 306. More importantly, our approach does not degrade too much even with a dictionary of size 1530. The computation time for learning a dictionary using these state-of-art approaches increases with the dictionary size; however, the computation time for our approach remains nearly constant.

5.2. Dictionary Learning from A Large Set of Dictionary Item Candidates

5.2.1 Keck Gesture Dataset

The Keck gesture dataset [17] consists of 14 different gestures [4], which are a subset of the military signals. We use the silhouette-based descriptor from [17] to capture shape information while we use optical-flow based features to encode motion information. Both types of feature descriptor are 256 dimensions. We compute the class distribution matrix introduced in [26] to obtain the count matrix $\mathcal{N}$.

[4]The gesture classes include turn left, turn right, attention left, attention right, flap, stop left, stop right, stop both, attention both, start, go back, close distance, speed up and come near.

Table 3. Computation time (s) for dictionary training using spatial pyramid features on the Caltech101 dataset.

Dict. size	306	510	714	918	1122	1326	1530
SDL	37.5	36.7	36.6	36.9	37.1	36.7	36.7
K-SVD [1]	578.3	790.1	1055	1337	1665	2110	2467
D-KSVD [35]	560.1	801.3	1061	1355	1696	2081	2551
LC-KSVD [12]	612.1	880.6	1182	1543	1971	2496	3112

Table 4. Computation time (s) for dictionary training using shape features on the Keck gesture dataset.

Dict. size	40	60	80	100	120	140	160	180
SDL	1.0	1.0	1.1	1.0	1.0	1.1	1.0	1.0
K-means	1.2	1.1	1.6	1.4	1.8	2.1	2.1	2.2
ME [10]	48.5	57.2	70.2	84.6	91.5	113.1	118.9	130
LiuShah [18]	599.2	597.9	597.2	596.1	593.9	590.3	587.4	582
MMI [26]	64.6	92.6	115.5	140.3	150.1	164.1	184.4	201

Following the experimental settings in [26], we obtain an initial large dictionary D^0 via K-SVD. The dictionary size $|D^0|$ is set to be approximately twice the dimension of the feature descriptor. Given D^0, our aim is to learn a compact and discriminative dictionary D^*.

We employ a leave-one-person-out protocol to report the classification result. A sparsity value of 30 is used. We evaluate our approach with different dictionary sizes and different features. Then we compare our results with the baseline, K-means, and *Liu-Shah* [18], ME [10] and MMI [26]. In Figure 7, our results outperform the baseline, K-means, *Liu-Shah*, ME and MMI, and is comparable to MMI* [5].

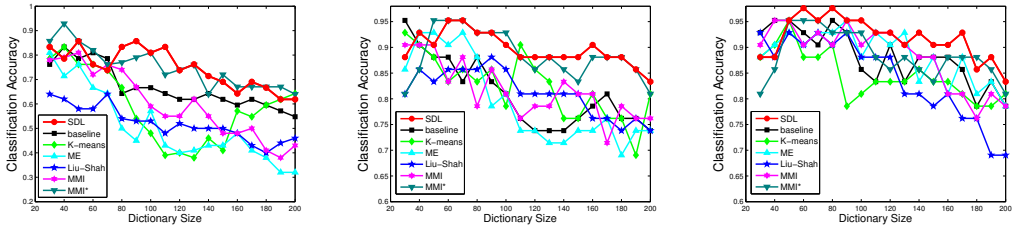
To evaluate the discrimination and compactness of learned dictionaries, we learned a 40 element dictionary from D^0 using six different approaches. Purity is the histogram of maximum probabilities of observing any class given a dictionary item [14]. Compactness is captured by the histogram of pairwise correlation coefficients of items in D^* [35] (lower correlations better). From Figure 8, our approach is most pure, and second most compact compared to the baseline, K-means, and *Liu-Shah*, ME and MMI.

In addition, we compare the computation time to learn a dictionary using shape features with K-means, *Liu-Shah*, ME and MMI. The parameter k is set to 3. Our approach is at least 50 times faster than *Liu-Shah*, ME and MMI in Table 4. The computation time using K-means is comparable to our approach, but its classification accuracy performance is much poorer than ours, as shown in Figure 7.

6. Conclusion

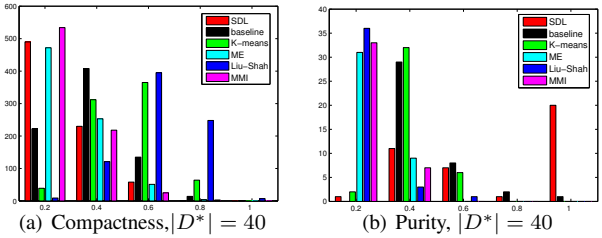
We present a greedy-based dictionary learning approach via maximizing a monotonically increasing and submodular function. By integrating the entropy rate of a random walk on a graph and a discriminative term into the

[5]MMI and MMI* here are the MMI1 and MMI2 approaches in [26] respectively. K-means, ME and MMI results are based on our own implementations while MMI* results are copied from the original paper. For the K-means method, we perform K-means clustering over D^0 to obtain D^*.



(a) Shape, $|D^0| = 600, k = 3$ (b) Motion, $|D^0| = 600, k = 6$ (c) Shape + Motion, $|D^0| = 1200, k = 2$

Figure 7. Classification accuracy using different approaches with different features and dictionary sizes. The results for MMI* are copied from [26]. The classification accuracy using initial dictionary D^0: (1) 26% (shape); (2) 26% (motion); (3) 36% (joint shape-motion).



(a) Compactness,$|D^*| = 40$ (b) Purity, $|D^*| = 40$

Figure 8. Purity and compactness comparisons with dictionary size 40 on the Keck gesture dataset. At the left-most bin of (a) and the right-most bin of (b), a compact and discriminative dictionary should have high purity and high compactness.

objective function, which makes each cluster compact and class pure, the learned dictionary is both representative and discriminative. The objective function is optimized by a highly efficient greedy algorithm, which can be easily applied to a large dataset. It outperforms recently proposed dictionary learning approaches including D-KSVD [35], SRC [30], LLC [29], *Liu-Shah* [18], ME [10], MMI [26], and can be comparable to LC-KSVD [12]. Possible future work includes exploring methods for efficiently updating the learned dictionary for a new category while keeping its the representative and discriminative power.

Acknowledgement

This work was supported by the Army Research Office MURI Grant W911NF-09-1-0383 and the Global Land Cover Facility with NASA Grants NNX08AP33A, NNX08AN72G and NNX11AH67G.

References

[1] M. Aharon, M. Elad, and A. Bruckstein. K-svd: An algorithm for designing overcomplete dictionries for sparse representation. *IEEE Trans. on Signal Processing*, 54(1):4311–4322, 2006.

[2] O. Boiman, E. Shechtman, and M. Irani. In defense of nearest-neighor based image classification, 2008. *CVPR*.

[3] T. Cormen, C. Leiserson, R. Rivest, and C. Stein. *Introduction to Algorithms, 3rd Edition*, 2009.

[4] T. Cover and J. Thomas. *Elements of Information Theory, 2nd Edition*, 2006.

[5] K. Engan, S. Aase, and J. Husϕy. Frame based signal compression using method of optimal directions (mod), 1999. *IEEE Symp. Circ. Syst.*

[6] L. FeiFei, R. Fergus, and P. Perona. Learning generative visual models from few training samples, 2004. *CVPR Workshop*.

[7] J. Gemert, J. Geusebroek, C. Veenman, and A. Smeulders. Kernel codebooks for scene categorization, 2008. *ECCV*.

[8] A. Georghiades, P. Belhumeur, and D. Kriegman. Illumination cone models for face recognition under variable lighting and pose. *TPAMI*, 23(6):643–660, 2001.

[9] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset, 2007. *CIT Technical Report 7694*.

[10] C. Guestrin, A. Krause, and A. Singh. Near-optimal sensor placements in gaussian processes, 2005. *ICML*.

[11] P. Jain, B. Kullis, and K. Grauman. Fast image search for learned metrics, 2008. *CVPR*.

[12] Z. Jiang, Z. Lin, and L. Davis. Learning a discriminative dictionary for sparse coding via label consistent k-svd, 2011. *CVPR*.

[13] A. Krause and V. Cevher. Submodular dictionary selection for sparse representation, 2010. *ICML*.

[14] S. Lazebnik and M. Raginsky. Supervised learning of quantizer codebooks by information loss minimization. *TPAMI*, 31(7):1294–1309, 2009.

[15] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, 2007. *CVPR*.

[16] H. Lee, A. Battle, R. Raina, and A. Y. Ng. Efficient sparse coding algorithms, 2006. *NIPS*.

[17] Z. Lin, Z. Jiang, and L. Davis. Recognizing actions by shape-motion prototypes trees, 2009. *ICCV*.

[18] J. Liu and M. Shah. Learning human actions via information maximization, 2008. *CVPR*.

[19] M. Liu, O. Tuzel, S. Ramalingam, and R. Chellappa. Entropy rate superpixel segmentation, 2011. *CVPR*.

[20] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Discriminative learned dictionaries for local image analysis, 2008. *CVPR*.

[21] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Supervised dictionary learning, 2009. *NIPS*.

[22] J. Marial, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding, 2009. *ICML*.

[23] M. Narasimhan, N. Jojic, and J. Bilmes. Q-clustering, 2005. *NIPS*.

[24] G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of the approximations for maximizing submodular set functions. *Mathematical Programming*, 1978.

[25] D. Pham and S. Venkatesh. Joint learning and dictionary construction for pattern recognition, 2008. *CVPR*.

[26] Q. Qiu, Z. Jiang, and R. Chellappa. Sparse dictionary-based representation and recognition of action attributes, 2011. *ICCV*.

[27] F. Rodriguez and G. Sapiro. Sparse representations for image classification: Learning discriminative and reconstructive non-parametric dictionaries.

[28] C. Rother, V. Kolmogorov, and A. Blake. Interactive foreground extraction using iterated graph cuts, 2004. *SIGGRAPH*.

[29] J. Wang, J. Yang, K. Yu, F. Lv, T. huang, and Y. Gong. Locality-constrained linear coding for image classification, 2010. *CVPR*.

[30] J. Wright, M. Yang, A. Ganesh, S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *TPAMI*, 31(2):210–227, 2009.

[31] J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification, 2009. *CVPR*.

[32] J. Yang, K. Yu, and T. Huang. Supervised translation-invariant sparse coding, 2010. *CVPR*.

[33] L. Yang, R. Jin, R. Sukthankar, and F. Jurie. Unifying discriminative visual codebook genearation with classifier training for object category recognition, 2008. *CVPR*.

[34] H. Zhang, A. Berg, M. Maire, and J. Malik. Svm-knn: Discriminative nearest neighbor classification for visual category recognition, 2006. *CVPR*.

[35] Q. Zhang and B. Li. Discriminative k-svd for dictionary learning in face recognition, 2010. *CVPR*.

[36] L. Zhao, H. Nagamochi, and T. Ibaraki. Greedy splitting algorithms for approximating multiway partition problems. *Mathematical Programming*, 2005.