

# Exemplar Based Deep Discriminative and Shareable Feature Learning for Scene Image Classification

Zhen Zuo<sup>a</sup>, Gang Wang<sup>a,b,\*</sup>, Bing Shuai<sup>a</sup>, Lifan Zhao<sup>a</sup>, Qingxiong Yang<sup>c</sup>

<sup>a</sup>*Nanyang Technological University, Singapore*

<sup>b</sup>*Advanced Digital Sciences Center, Singapore*

<sup>c</sup>*City University of Hong Kong, Hong Kong*

---

## Abstract

In order to encode the class correlation and class specific information in image representation, we propose a new local feature learning approach named Deep Discriminative and Shareable Feature Learning (DDSFL). DDSFL aims to hierarchically learn feature transformation filter banks to transform raw pixel image patches to features. The learned filter banks are expected to: (1) encode common visual patterns of a flexible number of categories; (2) encode discriminative information; and (3) hierarchically extract patterns at different visual levels. Particularly, in each single layer of DDSFL, shareable filters are jointly learned for classes which share the similar patterns. Discriminative power of the filters is achieved by enforcing the features from the same category to be close, while features from different categories to be far away from each other. Furthermore, we also propose two exemplar selection methods to iteratively select training data for more efficient and effective learning. Based on the experimental results, DDSFL can achieve very promising performance, and it also shows great complementary effect to the state-of-the-art Caffe features.

**Keywords:** Deep Feature Learning, Information Sharing, Discriminative Training, Scene Image Classification

---

## 1. Introduction

Extracting informative, robust, and compact data representation (features) has been considered as one of the key factors for good performance in computer vision. Thus, much effort has been paid on developing efficient and effective features, and the existing methods can roughly be classified into two categories: feature engineering and feature learning. In the last decade, numerous feature engineering methods developed hand-crafted features, such as SIFT [1], and HOG [2], have ruled the image representation area. However, such methods are labor-intensive and limited by the designer's ingenuity and prior knowledge. In contrast, to expand the capability and ease of image representation, feature learning methods [3–9] aim to automatically learn data adaptive image representations from raw pixel image data. However, these methods are generally poor on extracting and organizing the discriminative information from the data. Meanwhile, most of the learning frameworks operate in unsupervised ways without considering the class label information, which is crucial for image classification. To compensate these weaknesses while maintain the advantages of feature learning, we propose to encode shareable information that exists among groups of classes, and discriminative patterns owned by specific classes in feature learning procedure.

In this paper, we develop a multiple layer feature learning framework called Deep Discriminative and Shareable Feature Learning (DDSFL), which aims to hierarchically learn transformation filter bank to transform pixel values of local image patches to features. As shown in Figure 1, in each feature learning layer,

---

\*Corresponding author

*Email addresses:* ZZU01@e.ntu.edu.sg (Zhen Zuo), wanggang@ntu.edu.sg (Gang Wang), BSHUAI001@e.ntu.edu.sg (Bing Shuai), ZHA00145@e.ntu.edu.sg (Lifan Zhao), qiyang@cityu.edu.hk (Qingxiong Yang)

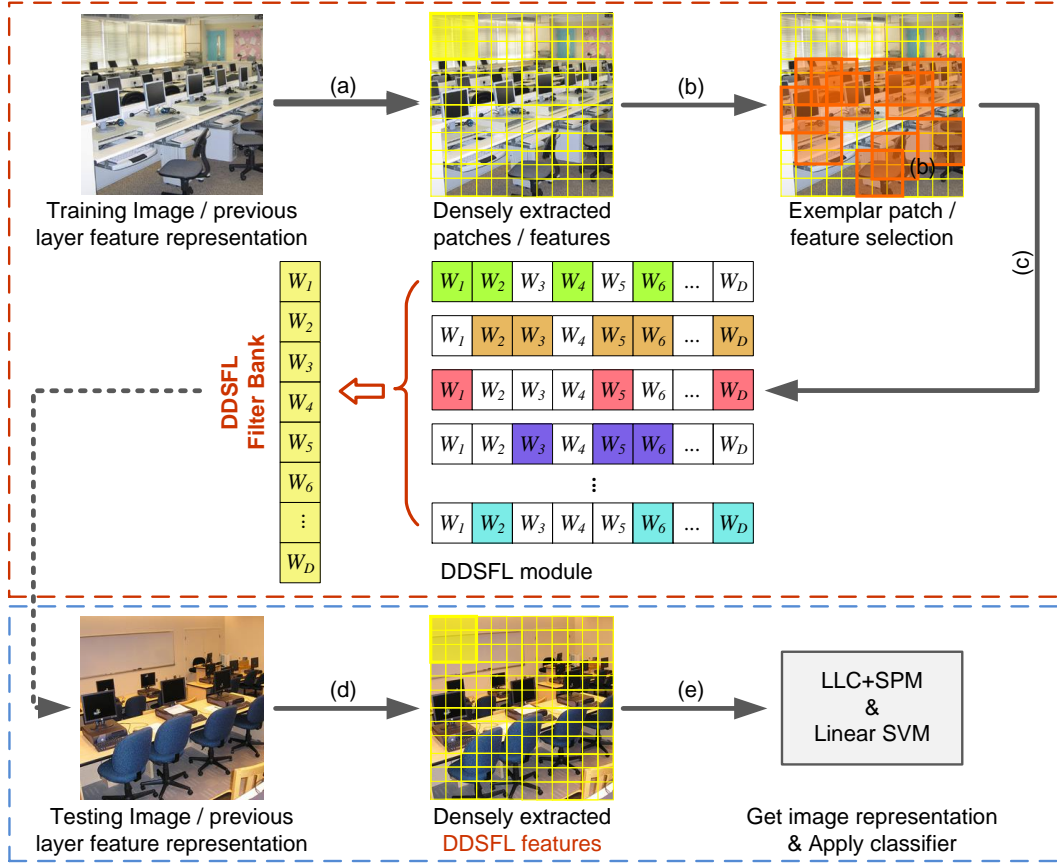


Figure 1: Illustration of single layer DDSFL. The red dashed box on the top describes the training procedure, while the blue dashed box below indicates the testing process. For the training procedure: (a) Input the original image or feature outputs of the previous layer (if there are any), and densely extract image patches or local features (yellow boxes); (b) Select exemplars (red boxes) for training; (c) Process DDSFL training module and learn filter bank. For the testing procedure: (d) Apply the learned filter bank  $W$  to the original input image or previous layer features, and densely extract the DDSFL feature for the current layer; (e) Process LLC [10] and SPM [11] afterwards to convert the local features to global image representation, and applying linear SVM to do final classification. In the DDSFL module,  $w_1, \dots, w_D$  represent the filters in the global filter bank  $W$ . During training step, for each class, we force it to activate a small subset of filters (the activated filters have been highlighted with different colors), and different classes can share the same filters. Finally, the DDSFL feature of a image patch  $x_i$  can be represented as  $f_i = \mathcal{F}(Wx_i)$ . (Best viewed in color)

we aim to learn an over-complete filter bank, which is able to cover the variances of patches from different classes, meanwhile keeping the shareable correlation among similar classes and discriminative power of each category. Intuitively, this goal can be reached by randomly selecting training patches, and learning filter banks for each class independently, then concatenating them together afterwards. However, there are several problems: (1) some of the patterns are shared among some classes, repeatedly learning filters corresponding to the similar patterns are neither memory compact nor computationally efficient, meanwhile the feature dimension will increase linearly with the number of classes, which limit the learning methods to be only applicable to small datasets; (2) discriminative power can hardly be fully exploited, since the class specific characteristics are generally subtle and not obvious without comparing with other classes; (3) in most cases, images are dominated by noisy or meaningless patches, learning filters from randomly sampled image patches will increase the learning cost and depress the performance.

To learn compact and effective filter banks, each category is forced to only activate a subset of the global filters during the learning procedure. Beyond reducing feature dimension, sharing filters can also lead to more robust features. Images belonging to different classes do share some information (e.g. in scene images, both

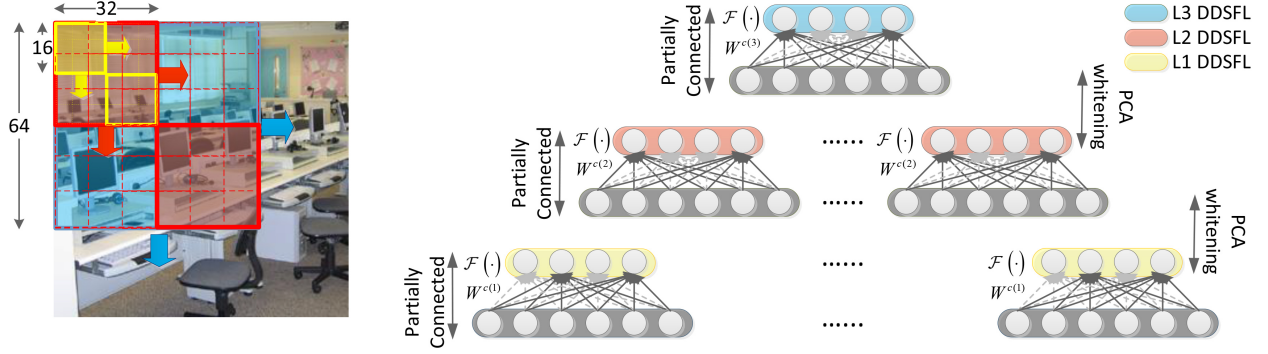


Figure 2: The hierarchical structure of DDSFL. In the input layer, 16x16 raw image patches are extracted, and be transformed to the first layer features with the first partially connected (in the testing step, fully connected) layer. Afterwards, combining these features correspond to 32x32 image areas, and applying PCA to prepare the reduced dimensional inputs to the second layer. Similarly, the third layer can be processed. Within each partially connected layer, the solid arrows indicate that the corresponding filters are activated for class  $c$ , while the gray dashed arrows mean filters are not activated. (Best viewed in color)

‘computer room’ and ‘office’ contain ‘computer’ and ‘desk’). The amount of information sharing depends on the similarity between different categories. Hence, we allow filters to be shareable, meaning that the same filters can be activated by a number of categories. We introduce a binary selection vector to adaptively select which filters to share, and among which categories.

To improve the discrimination power, we force the features from the same category to be close and the features from different categories to be far away (e.g. patches corresponding to bookshelf in ‘office’ can hardly be found in ‘computer room’). However, the local patches from the same categories are very diverse. Therefore, we propose to measure the similarity by forcing a patch to be similar to a subgroup of the training samples from the same category instead. Furthermore, not all the local patches from different classes need to be separable. Thus, we relax the discriminative term to allow sharing similar patches across different classes, and focus on separating the less similar patches.

To improve the quality of the filters and efficiency of the learning procedures, we propose two exemplar selection schemes to select effective training data. The proposed methods aim to remove the noisy training patches that commonly exist in many different classes, and select the patches that contain both shareable and discriminative patterns as the training data to learn the filter banks.

Furthermore, supported by lots of previous deep feature learning works, hierarchically extracting increasing visual level features can help to get more abstractive and useful information. Inspired by this idea, we extend the single layer feature learning module to a hierarchical structure. In this paper, we build a three layer learning framework, as shown in Figure 2. Specifically, we firstly learn the first layer features from small (16x16) raw pixel value image patches. Then for the higher layers, we convolve the previous layer features within a larger region (32x32 and 64x64 for the second and third layer respectively) as the input to PCA, and use the reduced dimensional data (we set the dimension to 300 for all the layers) as the input to train the current layer filter bank (we set to learn 400 filters for each layer). Finally, we combine the features learned by all the three layers as our DDSFL feature.

The rest of this paper is organized as follows. Section 2 introduces the related works, including feature engineering, feature learning, and discriminative training. Section 3 describes the details of our DDSFL method by introducing global unsupervised term, shareable term, and discriminative term. Section 4 proposes two exemplar selection methods including Nearest Neighbor based and SVM based selection. In Section 5, we test our method on three widely used scene image classification datasets: Scene 15, UIUC Sports, MIT Indoor, and we also test on PASCAL VOC 2012. The experimental results show that our features can outperform most of the existing methods, and it also has significant complementary effect with the state-of-the-art Caffe [12] features (ConvNets [3] pre-trained on ImageNet [13]). Finally, Section 6 concludes this paper.

## 2. Related Works

### 2.1. Feature Engineering

In the feature engineering area, hand-crafted features including SIFT [1], HOG [2], LBP [14] and GIST [15] (global feature) were popular used. Comparing to current existing feature learning methods, they can generally get better local descriptors, and extra information (e.g. discriminative information) can be better expressed by manually inserting prior knowledge. Even though they are very powerful, designing such features is labor intensive, and they can hardly capture any information other than what have been defined by the designers. Therefore, they can hardly extract high visual level patterns. In this paper, we aim to hierarchically learn a data adaptive feature representation.

### 2.2. Feature Learning

In the feature learning field, directly learning features from image pixel values [3–9, 16–21] emerges as a hot research topic in computer vision. These methods are able to learn data adaptive features, and they are usually easy to extend to hierarchical structure, and learn multiple levels of image representations. According to different training settings, there are usually two lines of works: unsupervised/semisupervised feature learning, and supervised feature learning.

Many of the current feature learning frameworks are unsupervised/semisupervised ones, these methods can usually generate numerous additional training data for different objectives, and utilize the intrinsic data similarities to learn transform invariant features. These methods have achieved superior performance on many important computer vision tasks such as image recognition [19, 22, 23], scene classification [20, 21], and action recognition [17]. Different from the unsupervised/semisupervised feature learning algorithms, where the class labels are ignored or weakly used, we argue that class specific discriminative information is critical for classification and discriminative patterns can be learned for better image representation. We experimentally show that our DDSFL performs better than unsupervised feature learning on scene datasets by utilizing the shareable and discriminative class correlations.

In the supervised feature learning line, the ConvNets [3] are the most popular deep feature learning structure, they focus on progressively learning multi-levels of visual patterns. ConvNets have been widely utilized to generate global image representations or mid-level discriminative features. They are the state-of-the-art feature learning frameworks on many tasks, such as image recognition [12, 22, 24–26], scene classification [27, 28], scene labeling [29], and contour detection [30]. In contrast, our DDSFL focuses on encoding the shareable and discriminative correlations among different classes in each layer’s feature transformation. Furthermore, different features usually capture different information, fusing complementary features will lead to better performance [23, 31]. Although many deep feature learning frameworks are very powerful, they generally focus more on high level image representation, while the low level features (first few layers) are relatively weak. While in this work, our DDSFL framework focus on encoding the class-level discriminative and shareable properties in patch-level local features, which will lead to better image representation for classification tasks. In Section 5, we will show that our DDSFL learns significant complementary information to the powerful ConvNets features, and combines with which, we can update the current state-of-the-arts on the Scene 15, UIUC Sports, MIT Indoor datasets, and get promising results on PASCAL VOC 2012.

### 2.3. Discriminative Training

There are also some previous works take discriminative information into consideration. For example, [32–35] learn discriminative dictionaries to encode local image features. Another line of works are usually known as mid-level feature searching or exemplar based representation [36–41]. They represent images in terms of weakly-supervise mined discriminative regions of interest (multi-scale patches, usually correspond to middle or high level patterns) or images. Afterwards, images can be represented with the max pooled responses of such mid-level patterns. In contrast, DDSFL focus on discriminatively learning filters, which transform local image patches to features, and allowing sharing local feature transformation filters between different categories. To the best of our knowledge, this hasn’t been done before. From another aspect,

compared with the previous exemplar based methods, our proposed exemplar selection methods are used to get more informative local patch level training data for the feature learning modules, thus the previous methods (middle or image level exemplar selection) are not suitable in this scenario. Furthermore, in [42], discriminative information is encoded by maximizing the margin between features from different classes, while making data from the same class to be close. Different from this work, we focus on learning local features for complicated data like scene images, where patches from the same class can be very different, while patches from different classes can be highly similar, thus, max-margin based discriminative learning cannot be directly used here. In [43, 44], object part filters at the middle level are shared to represent a large number of object categories for object detection. While we only use weakly supervised image level labels, and develop a nearest neighbor based maximum margin method to learn discriminative feature transformation matrix.

This paper is an extension of our published conference paper [45]. We have proposed a new fast exemplar selection method, provided more details about the previous exemplar selection method, given more details of the algorithm, and extended the previous feature learning framework to a three layer deeper one. We have also provided more complete experiment results: performance on PASCAL VOC 2012, the performance details of using different number of layers, and accuracy of using random initialed filter banks.

### 3. Deep Discriminative and Shareable Feature Learning

In this section, the hierarchical structure of our Deep Discriminative and Shareable Feature Learning (DDSFL) framework will be briefly introduced. For each single layer DDSFL, its three learning components will be introduced in detail, and an alternating optimization strategy will be provided afterwards.

#### 3.1. DDSFL Hierarchical Structure

In this paper,  $x_i \in \mathbb{R}^{D_0}$  is denoted as the vectorized raw pixel values of an image patch, and  $X = \{x_1, x_2, \dots, x_N\}$  refer to as the collection of all the  $N$  training patches densely extracted from the training images (the patches are weakly labeled by the image level class labels). As shown in Figure 2, there are  $L = 3$  layers in our DDSFL framework. In each layer of DDSFL, we aim to learn a feature transformation filter bank  $W^{(l)} \in \mathbb{R}^{D \times D_0}$ ,  $l = \{1, \dots, L\}$ , in which,  $l$  is the layer indicator,  $D_0$  denotes the dimension of the input data,  $D$  is the number of filters, and each row of  $W^{(l)}$  represents a filter. By multiplying  $W^{(l)}$  with an input vector  $x_i^{(l)}$ , and applying an activation function  $\mathcal{F}(\cdot)$ , we expect to generate feature  $f_i^{(l)} = \mathcal{F}(W^{(l)}x_i^{(l)})$ , which is discriminative and as compact as possible. Afterwards, by convolving the  $f_i^{(l)}$  in a larger image receptive field, and process PCA, we can get the input  $x_i^{(l+1)}$  of the next layer.

#### 3.2. Single Layer DDSFL Learning Components

In each single layer, our DDSFL framework aims to learn features that are able to: 1) preserve the information of the original data; 2) compactly represent patterns shared among similar classes; and 3) effectively encode class specific discriminative patterns. To achieve these goals, three corresponding learning components are proposed: the global unsupervised term, shareable pattern learning term, and discriminative information encoding term.

For the compact consideration, we force each class to activate a subset of filters in  $W^{(l)}$ , and using the training patches from the similar classes (classes activate the same filters) to train the shared filters. To encode discriminative information, we force the features from the same class to be similar, while features from different classes to be as different as possible. In the following subsections, we will introduce the details of the three learning components, and for the sake of convenience, we ignore the  $l$  in all the variables, and focus on the learning procedure of a single layer DDSFL.

### 3.2.1. The Global Unsupervised Term

To ensure the learned filter bank  $W$  is able to preserve the information in the original data after processing feature transformation, meanwhile, also be robust to the small variance exist in neighborhood spatial areas, we firstly introduced a global reconstruction term and build an auto-encoder like learning scheme:

$$\begin{aligned}
L_u &= \sum_{i=1}^N \mathcal{L}_r(x_i, W) + \xi \sum_{i=1}^N \mathcal{L}_{sp}(x_i, \Omega(x_i)) + \lambda_1 \sum_{i=1}^N \|f_i\|_1 \\
\text{where } \mathcal{L}_r(x_i, W) &= \|x_i - W^T W x_i\|_2^2 \\
\mathcal{L}_{sp}(x_i, \Omega(x_i)) &= \frac{1}{M} \sum_{m=1}^M \|f_i - \Omega_m(f_i)\|_1 \\
f_i &= \mathcal{F}(W x_i), \quad \mathcal{F}(\cdot) = \text{abs}(\cdot)
\end{aligned} \tag{1}$$

where  $\mathcal{L}_r$  is the global reconstruction error with respect to global filter bank  $W$  and training data  $x_i$ . This auto-encoder [4, 46] style reconstruction cost penalization term can not only prevent  $W$  from degeneration, but also allow  $W$  to be over-complete. The spatial denoising term  $\mathcal{L}_{sp}$  is used to minimize the difference between the training patch and its spatial neighbor patches, and makes the learned features to be robust to small spatial variance<sup>1</sup>. For each training data  $x_i$ , we randomly sample  $M$  patches (extract different size patches, and resize them to have the same size with  $x_i$ ) from the spatial neighborhood image areas, which are denoted as  $\Omega(x_i)$ . The term  $\|f_i\|_1$  is used to enforce the sparsity of the learned feature  $f_i$ . To make the optimization easier while also preserving the quality of the learned features, we use the non-negative activation function in [5, 17], and set  $\mathcal{F}(\cdot) = \text{abs}(\cdot)$ . Then the sparse term  $\|f_i\|_1$  degenerates to the summation of all the elements in vector  $f_i$ .

### 3.2.2. Shareable Pattern Learning Term

Although the above global unsupervised term can already lead to satisfactory results, class information has not been used, which is critical for classification problem. Furthermore, we aim to learn a compact filter bank, rather than a redundant one. Thus, we propose to encode the shareable information exists among similar classes, and utilize the training data from these classes to jointly learn the sharable filters.

To reach this goal, we introduce a sparse binary vector  $\alpha^c \in \mathbb{R}^D$  for each class  $c \in \{1, \dots, C\}$  ( $C$  is the number of classes) to indicate the selection status of rows of  $W$ . For example, if  $\alpha_d^c = 1$ ,  $d = 1, \dots, D$ , then the  $d$ -th row of  $W$  is activated. For each class, only a small number of filters can be activated, while the same filter can be potentially activated by several classes. The cost of our shareable constraint term of class  $c$  is formulated as following:

$$\begin{aligned}
L_{\text{sha}}^c &= \sum_{j=1}^{N_c} \mathcal{L}_{\text{sha}}^c(x_j^c, W^c) + \lambda_2 \|\alpha^c\|_0 \\
\text{s.t. } \alpha_d^c &\in \{0, 1\}, \quad d = 1, \dots, D \quad W^c = \text{diag}(\alpha^c) W \\
\text{where } \mathcal{L}_{\text{sha}}^c(x_j^c, W^c) &= \|x_j^c - (W^c)^T W^c x_j^c\|_2^2
\end{aligned} \tag{2}$$

where  $W^c$  is the selected filters of class  $c$ , and  $N_c$  is the number of training patches from class  $c$ . Similar to  $\mathcal{L}_u$ ,  $\mathcal{L}_{\text{sha}}^c$  is the reconstruction cost function with respect to  $W^c$  and training patch  $x_j^c$  from class  $c$ . The  $\|\alpha^c\|_0$  term is used to force the sparsity, consequently, only a small number of rows of  $W$  will be activated. Here we apply a greedy method to search for appropriate filters to activate. When  $\alpha^c$  is updated in each iteration, the corresponding training data for each filter will also be updated.

<sup>1</sup>According to our experiment results, adding the spatial denoising term will not bring much performance gain, but it can help the learning result be more robust to different initialization.

---

**Algorithm 1:** Deep Discriminative and Shareable Feature Learning

---

**Input:** $x_i$ : Unlabeled training patch $x_j^c$ : Image-level labeled training patch from class  $c$  $D$ : Number of filters in the global filter bank $L$ : Number of feature transformation layers $C$ : Number of the classes $\xi, \gamma, \eta, \lambda_1, \lambda_2$ : Trade off parameters for controlling weight of spatial denoising term, shareable term, discriminative term, and sparsity**Output:** $W^{(l)}$ : Global filter bank (feature transformation matrix) of layer  $l$ 1. Initialize  $\alpha^c = \mathbf{0}^T$ 2. Set  $W^{(l)}$  as a random number  $D \times D_0$  matrix3. Learn  $W^{(l)}$  with  $L_u$  as the initialized  $W^{(l)}$  to the DDSFL

4. Select training exemplars for each class (Section 4)

5. Search the positive and negative nearest neighbor sets for each  $x_j^c$ **for**  $l = 1$  to  $L$  **do**    **while**  $W^{(l)}$  and  $\alpha^c$  not converge **do**        **for**  $c = 1$  to  $C$  **do**            6. Fix  $W^{(l)}$  and all the  $\alpha^c$ , solve Equation 6 by updating  $\alpha^c$         **end**        7. Fix  $\alpha^c, c = 1, \dots, C$  and solve Equation 5 by updating  $W^{(l)}$     **end**    **return**  $W^{(l)}$ **end**

---

### 3.2.3. Discriminative Information Encoding Term

To enhance the discriminative power of the features, we further introduce a term based on the assumption that discriminative features should be close to the features from the same category, and be far away from the features from the different categories. However, patches from the same image class are inherently diverse. If we get a triplet  $\{x_i, x_i^+, x_i^-\}$ , in which,  $x_i$  denotes the training patch,  $x_i^+$  denotes a positive patch randomly select from the same class, and  $x_i^-$  represents a negative patch from the other classes, it is highly possible that  $x_i$  is similar to  $x_i^-$  rather than  $x_i^+$ . To solve this problem, we adopt the nearest neighbor based ‘patch-to-class’ distance metric [47–49]. For a training patch  $x_j^c$ , its positive nearest neighbor patch set from the same category is denoted as  $\Gamma(x_j^c)$ ; and its negative nearest neighbor patch set from the categories other than  $c$  is denoted as  $\bar{\Gamma}(x_j^c)$ . The  $k$ -th nearest neighbor in the two sets are represented as  $\Gamma_k(x_j^c)$  and  $\bar{\Gamma}_k(x_j^c)$  respectively.

In the class-specific feature space of class  $c$  (transformed by  $W^c$ ), the feature representation of the  $k$ -th positive and negative nearest neighbor patches are denoted as  $\Gamma_k(f_j^c) = \mathcal{F}(W^c \Gamma_k(x_j^c))$  and  $\bar{\Gamma}_k(f_j^c) = \mathcal{F}(W^c \bar{\Gamma}_k(x_j^c))$  correspondingly. Then we get triplet  $\{f_j^c, \Gamma_k(f_j^c), \bar{\Gamma}_k(f_j^c)\}$ , and we aim to minimize the distance between  $f_j^c$  and  $\Gamma_k(f_j^c)$ , while maximize the distance between  $f_j^c$  and  $\bar{\Gamma}_k(f_j^c)$ . Furthermore, to make the learning procedure more efficient, we should focus on the ‘hard’ training samples (based on the

maximum margin theory in learning). Hence, we develop a ‘hinge-loss’ like objective function:

$$\begin{aligned}
L_{\text{dis}}^c &= \sum_{j=1}^{N_c} \max(\delta + \text{Dis}(x_j^c, \Gamma(x_j^c)) - \text{Dis}(x_j^c, \bar{\Gamma}(x_j^c)), 0) \\
\text{where } \text{Dis}(x_j^c, \Gamma(x_j^c)) &= \frac{1}{K} \sum_{k=1}^K \|f_j^c - \Gamma_k(f_j^c)\|_2^2 \\
\text{Dis}(x_j^c, \bar{\Gamma}(x_j^c)) &= \frac{1}{K} \sum_{k=1}^K \|f_j^c - \bar{\Gamma}_k(f_j^c)\|_2^2
\end{aligned} \tag{3}$$

where  $K$  is the number of nearest neighbors in the nearest neighbor patch sets, we fixed it as 5, and  $\delta$  is the margin, we set it to 1 in our experiments.

Furthermore, the data in the triplet  $\{x_j^c, \Gamma_k(x_j^c), \bar{\Gamma}_k(x_j^c)\}$  depends on the values of  $W^c$ , which is updating during each iteration. Consequently, when  $W^c$  is iteratively being optimized and updated, the nearest neighbors will also be different. Since the nearest neighbor searching is time consuming, we assume the nearest neighbor members  $\Gamma_k(x_j^c)$  and  $\bar{\Gamma}_k(x_j^c)$  (the original image patches correspond to the triplet in the feature space) will keep unchanged when  $W^c$  does not change too much, and we only update the nearest neighbor members every 50 iterations of  $W^c$  updating.

### 3.3. DDSFL Objective Function and Optimization

Combing the above three learning components, the complete DDSFL objective function can be expressed as:

$$\begin{aligned}
&\min_{W, \alpha^c} L_u + \gamma \sum_{c=1}^C L_{\text{sha}}^c + \eta \sum_{c=1}^C L_{\text{dis}}^c \\
\text{where } L_u &= \sum_{i=1}^N \mathcal{L}_r(x_i, W) + \xi \sum_{i=1}^N \mathcal{L}_{sp}(x_i, \Omega(x_i)) + \lambda_1 \sum_{i=1}^N \|f_i\|_1 \\
L_{\text{sha}}^c &= \sum_{j=1}^{N_c} \mathcal{L}_{\text{sha}}^c(x_j^c, W^c) + \lambda_2 \|\alpha^c\|_0 \\
L_{\text{dis}}^c &= \sum_{j=1}^{N_c} \max(\delta + \text{Dis}(x_j^c, \Gamma(x_j^c)) - \text{Dis}(x_j^c, \bar{\Gamma}(x_j^c)), 0) \\
\text{s.t. } &\alpha_d^c \in \{0, 1\}, d = 1, \dots, D \quad W^c = \text{diag}(\alpha^c) W
\end{aligned} \tag{4}$$

In Equation 4, we need to simultaneously optimize the global filter transformation matrix  $W$  and the class specific filter selection vector  $\alpha^c$ . This function cannot be jointly optimized with respect to  $W$  and  $\alpha^c$ . However, if one of them is fixed, the objective function turns to be convex. Therefore, we adopt an alternating optimization strategy to iteratively update  $W$  and  $\alpha^c$ .

We firstly fix  $\alpha^c$  for each class, and focus on updating  $W$ :

$$\begin{aligned}
&\min_W \sum_{i=1}^N \mathcal{L}_r(x_i, W) + \xi \sum_{i=1}^N \mathcal{L}_{sp}(x_i, \Omega(x_i)) + \lambda_1 \sum_{i=1}^N \|f_i\|_1 \\
&\quad + \gamma \sum_{c=1}^C \sum_{j=1}^{N_c} \mathcal{L}_{\text{sha}}^c(x_j^c, W^c) + \eta \sum_{c=1}^C L_{\text{dis}}^c
\end{aligned} \tag{5}$$

Because of the non-negative constraint applied on  $f_i$  (Section 3.2.1),  $\|f_i\|_1$  degenerates to summation of different elements in  $f_i$ . Consequently, Equation 5 can be easily optimized by unconstrained solvers like L-BFGS.



Next we fix the  $W$ , and focus on updating  $\alpha^c$  class by class:

$$\min_{\alpha^c} \sum_{j=1}^{N_c} \mathcal{L}_{\text{sha}}^c(x_j^c, W^c) + \lambda_2 \|\alpha^c\|_0 + \eta L_{\text{dis}}^c \quad (6)$$

We update one  $\alpha^c$  each time for the  $c$ -th class, and keep  $\alpha^{\bar{c}}$  ( $\bar{c} \neq c$ ) unchanged. To get  $\alpha^c$ , we apply a greedy optimization method. We first set all the elements in  $\alpha^c$  as 0, then we search for the single best filter that can minimize Equation 6, and activate that filter by setting the corresponding element in  $\alpha^c$  to 1. Afterwards, based on the previously activated filters, we search for next filter that can further minimize the cost function. The optimization terminates when the loss  $\mathcal{L}_{\text{sha}}^c$  is smaller than a threshold, and the renewed  $\alpha^c$  will be sent as the input to Equation 5 again to further optimize  $W$ .

The learning algorithm and initialization procedure are shown in Algorithm 1. The alternative optimization terminated until the values of both  $W$  and  $\alpha^c$  converge (takes about 5 rounds).

#### 4. Exemplar Selection

The quality of the learned features not only depends on the learning structure and parameters, but also relies on the quality of input training data. In order to select training patch set that carries both potential shareable and discriminative patterns, while also excludes common noisy patches, a training data selection procedure is required before processing feature learning.

For this purpose, one intuitive solution is applying k-means clustering, and using the cluster centroids as the exemplars [49] for training. However, as analyzed in [47], informative descriptors have low database frequency, and the conventional clustering methods prefer dominant patterns as inlier of clusters. Consequently, the non-informative patches (e.g. sky areas) are more likely to be selected rather than the discriminative ones. Thus, we propose two discriminative exemplar training data selection approaches: Nearest Neighbor (NN) based and Support Vector Machines (SVM) based exemplar selection.

##### 4.1. NN Based Exemplar Selection

Inspired by the image-level exemplar selection method in [40], we propose a NN based exemplar selection scheme to remove the patches that are likely to commonly exist in many classes.

In [40], the exemplar images are defined as the minimum set of ‘dominating images’ which can cover all the intra-class variations, meanwhile as distinctive as possible to the exemplars from other classes. Different from this scenario, our exemplars are local image patches/features. Furthermore, we aim to keep shareable patterns, while also select class-specific discriminative patterns. To make the processing more applicable, we manipulate the selection procedure by removing the uninformative common patterns existing in most of the classes.

Firstly, we define the ‘coverage set’ of a patch  $x_i$ . Given  $X$  as the original global patch set, which contains patches densely extracted from all the training images. For each patch  $x_i \in X$ , we search its  $M_{nn}$  nearest neighbors from  $X$ , and define these  $M_{nn}$  patches as the ‘coverage set’ of  $x_i$ .

Secondly, we define the ‘reaching number’ and ‘reaching distance’ of a patch  $x_j^c$ . If  $x_j^c$  is included in the ‘coverage set’ of a patch  $x_i^{\bar{c}}$  from classes other than  $c$ , it is ‘reached’ once, and the times of being reached over the patch set  $X^{\bar{c}}$  (from class  $\bar{c}$ ) is the ‘reaching number’ of  $x_j^c$ . Meanwhile, we also record the distances between  $x_j^c$  and the patches which cover  $x_j^c$ . The summation of these distances is the ‘reaching distance’ of  $x_j^c$ .

Intuitively, the larger the ‘reaching number’ or the smaller the ‘reaching distance’, the more common the  $x_j^c$ . However, if we only consider ‘reaching distance’, for the common patterns (e.g. sky area patches), there are numerous similar patches with small variation. Consequently, because of the limited size of ‘coverage set’, the chance that these common patches be ‘reached’ by the other similar common patches are not always high, and a common patch might get small ‘reaching number’, even though its similar patterns widely exist. On the other hand, if we only consider ‘reaching distance’, for a target discriminative patch, it might be only covered once, while the corresponding patches are highly similar to it, then its ‘reaching distance’ is

---

**Algorithm 2:** NN based Exemplar Selection

---

**Input:** $X$ : Global patch set $X_c$ : Patch set of class  $c$  $C$ : Number of the classes $M_{nn}$ : Number of patches in each coverage set $\varepsilon_{nn}$ : Threshold for selecting discriminative exemplars**Output:** $E_c$ : Exemplars of class  $c$ 1. Calculate the coverage set of each patch from  $X$ **for**  $c = 1$  to  $C$  **do**    2. For each patch from  $X_c$ , calculate its reaching score  $R_S(x_j^c)$  based on Equation 8    3. Descendingly sort the patches from  $X_c$  based on the reaching scores.    4. Select the top  $\varepsilon_{nn}$  percent ranked patches as the exemplars  $E_c$ **end****return**  $E_c$ 

---

unfairly small. Therefore, we make use of both ‘reaching number’ and ‘reaching distance’, and define the ‘reaching score’ of  $x_j^c$  to class  $\bar{c}$  as following:

$$R_S^{\bar{c}}(x_j^c, \bar{c}) = \frac{1}{R_N(\bar{c})} \sum_{l=1}^{R_N(\bar{c})} R_d \quad (7)$$

where  $R_d = \|x_j^c - x_l^{\bar{c}}\|_2$

where  $R_S^{\bar{c}}$  is the class wise reaching score,  $R_N(\bar{c})$  is ‘reaching number’ denotes the number of the patches from classes  $\bar{c}$  whose coverage sets contain  $x_j^c$ , and summation of  $R_d$  is the ‘reaching distance’.

Furthermore, to keep the shareable patterns, we also prefer to penalize the patterns appearing in most of the classes rather than shared among few classes. Thus, we define the final ‘reaching score’ as:

$$R_S(x_j^c) = \frac{1}{C-1} \sum_{\bar{c} \neq c} R_S^{\bar{c}}(x_j^c, \bar{c}) \quad (8)$$

where  $C$  is the number of classes. If  $x_j^c$  has low ‘reaching score’, then it means that  $x_j^c$  represents a common pattern, and should be removed in the training procedure, otherwise,  $x_j^c$  should be kept an exemplar.

For each class, we sort the patches based on their  $R_S(x_j^c)$  scores descendingly, and select the top  $\varepsilon_{nn}$  percent of them as discriminative exemplars. The selecting procedures are shown in Algorithm 2.

#### 4.2. SVM Based Exemplar Selection

Inspired by the mid-level discriminative exemplar searching methods [37, 50], we also try to use learning based methods to search for local patch level exemplars. Compare to the NN based selection method, SVM based method is much faster and cost less memory with small performance drop.

The desired exemplars should be both representative and discriminative: similar patches of the exemplars should be able to be found from the same or similar classes, and the exemplars of a certain class should also contain class-specific patterns. To reach these goals, we model the problem as a discriminative clustering problem, and the centers of the effective clusters are considered as the exemplars. In this paper, we use linear SVM to iteratively refine the selection planes for each cluster. In one iteration of each cluster, there are mainly two steps: SVM training and SVM testing (re-clustering). In the SVM training step, the patches from the same cluster are positive samples, and the patches from all the other classes are negative samples, and a linear SVM model is learned based on them. In the SVM testing step, the previous learned SVM

---

**Algorithm 3: SVM based Exemplar Selection**

---

**Input:**

$X^{tr}$ : Training patch set  
 $X_{cs}^{tr}$ : The  $s$ -th cluster of the training patch set from class  $c$   
 $X_{\bar{c}}^{tr}$ : Training patches from classes other than  $c$   
 $X^{val}$ : Validation patch set  
 $S_c$ : Number of initial clusters for each class  $c$   
 $C$ : Number of the classes  
 $M_{svm}$ : Maximum number of patches in each cluster

**Output:**

$E_c$ : Exemplars of class  $c$

1. Applying k-means to initialize  $S_c$  clusters for each class

**while** not converged **do**

**for**  $c = 1$  to  $C$  **do**

**for**  $s = 1$  to  $S_c$  **do**

            2. For each cluster  $s$  from class  $c$ , set the cluster members  $X_{cs}^{tr}$  as positive data and randomly sample patches from  $X_{\bar{c}}^{tr}$  as negative data, and train a linear SVM  
            3. Test the learned SVM on the validation set  $X^{val}$ . If the SVM meets any of the non-discriminative condition, remove the cluster.

**end**

**end**

4. Swap training set  $X^{tr}$  and validation set  $X^{val}$ .

**end**

5. Use the remaining qualified cluster centers as the exemplars  $E_c$

**return**  $E_c$

---

model is used to test on all the validation data patches (not visible in the training procedure), the patches which get fired<sup>2</sup> are kept and be used to replace the data in the original cluster, and they will be used to learn the new SVM model in the next iteration. To ensure the exemplars are representative to the class, we do not consider clusters with less than  $\varepsilon_{svm}$  ( $\varepsilon_{svm} = 3$ ) members. To ensure the purity of the clusters, we constrain that each effective cluster can have at most  $M_{svm}$  (we set  $M$  as 10) members.

We randomly pick a large subset of local patches from the training images as exemplar candidates, and run k-means clustering for all of the candidate patches of each class to initialize the clusters. Since the SVM heavily prefers the training data during testing, thus we equally separate the candidate patch set into training set and validation set. For the learned SVMs, we test them on the validation set class by class. If either of the following two conditions is true, then the cluster is considered as not discriminative, and should be removed:

- The SVM fires<sup>2</sup> less than  $\varepsilon_{svm}$  (3 in our experiment) patches from the same class (considered as outliers or noisy data);
- The SVM fires<sup>2</sup> many patches from more than half of all the classes (considered as common patterns exist everywhere).

Afterwards, we swap validation set and training set, and repeat the above procedures until convergence. The iteration usually terminates in about 5 rounds. The overall procedure is shown in Algorithm 3.

#### 4.3. NN Based vs SVM Based exemplar selection

We propose NN based and SVM based exemplar selection methods. According to the experiment results (Table 4), both of them bring significant improvements than DDSFL without exemplar selection procedure.

---

<sup>2</sup>Classification score higher than -1

Here we make a brief comparison between these two methods in three aspects: performance, computation cost, and memory cost.

In classification performance aspect, comparing with the result of randomly select training patch, NN based exemplar selection leads to around 1% to 3% improvements, while SVM based selection gets around 0.5-1.5% improvements. Thus, NN based method is a better choice for higher accuracy results.

Next we compare the two methods in computation cost aspect. For the NN based selection, the most time consuming procedure is the step 1 in Algorithm 2: coverage set searching. To speed up the procedure, we use the FLANN [51] toolbox, which is a library utilize multiple randomized k-d trees to achieve fast NN approximation for high dimensional data. For the SVM based selection, step 2 and 3 in Algorithm 3 (SVM training and testing) are the most and second most time consuming steps. To speed up the SVM training procedure, we apply stochastic learning of SVM [52]. While in the SVM testing step, for each trained linear SVM model, only classification score of the validation data patches need to be calculated. This procedure can be easily parallel computed, and we use GPU to speed up. In our experiments, the SVM based methods cost much less hours, especially when the number of training patches is very large.

At last, the memory costs are compared. In the NN based selection, we need to keep all the training patches  $X$  in the memory to apply coverage set searching for each patch. To depress the memory cost, we separate the  $X$  into several subgroups, process NN searching for each subgroup, and resort the searching results from all the subgroups afterwards. However, the more subgroups we generate, the longer overall NN searching time will be cost. On the contrary, SVM based selection requires very limited memory, and only an ignorable number of data points need to be loaded into the memory.

Therefore, NN based method can be used to achieve higher accuracy, while SVM based method can be manipulated faster with much less memory cost with less performance gain.

## 5. Experiments and Analysis

### 5.1. Datasets and Experiment Settings

We tested our DDSFL method on three widely used scene image classification datasets: Scene 15 [11], UIUC Sports [53], and MIT Indoor [54]. We also tested on the challenging PASCAL VOC 2012 object classification dataset. To make a fair comparison with other types of features, we only utilized gray scale information for all of the images in these datasets.

- **Scene 15:** This dataset includes 4,485 images from 15 outdoor and indoor scene categories, each category contains 200 to 400 gray scale images. Based on the standard setting, we use 100 images per category for training, and the rest images for testing.
- **UIUC Sports:** This dataset contains 1,579 from 8 sports event classes, each category contains 137 to 250 images. We use 70 images per class as training images, and 60 images per class as testing images.
- **MIT Indoor:** This dataset have 15,620 image in total, and they are from 67 indoor scene categories. Based on the splitting set in [54], there are around 80 training images and 20 testing images for each category.
- **PASCAL VOC 2012:** This challenge aims to recognize objects from a number of visual object classes in realistic scenes. In this dataset, 20 object classes are selected. There are 5,717 training images with 13,609 objects, 5,823 validation images with 13,841 objects. Thus, the total number of labeled images are 11,540, and we did not use any extra training data. In the testing procedure, 10,991 unlabeled testing images are provided, and the classification results can be achieved by uploading the classification scores to the PASCAL VOC evaluation server.

Since the resolution of the original images in UIUC Sports, MIT Indoor and PASCAL VOC 2012 are too high for learning local features efficiently, we resize them to have maximum 300 pixels along the longer axis. For Scene 15 and UIUC sports, we randomly split the training and testing datasets for 5 times, and

Methods	Scene 15	UIUC Sports	MIT Indoor
GIST[15]	73.28%	-	22.00%
CENTRIST [56]	83.10%	78.50%	36.90%
SIFT[1]	82.06%	85.12%	45.86%
HOG2x2[55]	81.58%	83.96%	43.76%
LBP[14]	82.95%	80.04%	39.25%
OTC[57]	84.37%	-	47.33%
Random <sup>3</sup>	70.91%	67.92%	21.34%
RICA[4]	80.01%	82.45%	47.76%
DDSFL	84.42%	86.91%	52.26%
Caffe[12]	87.99%	93.96%	58.52%
SIFT+ Caffe	89.90%	95.05%	70.51%
DDSFL + Caffe	<b>92.81%</b>	<b>96.78%</b>	<b>76.23%</b>

Table 1: Comparison results between our DDSFL feature and other features. (Caffe is the global feature learned by the deep ConvNets pre-trained on ImageNet.)

get the results of different methods based on the same splits. The average accuracy numbers over these 5 rounds are reported for comparison.

For all the local features, we densely extracted features from  $S$  scales ( $S$  equals to 6, 5, and 3 for the first, second, and third layer respectively) with rescaling factors  $2^{-i/2}, i = 0, 1, \dots, S$ . Specifically, Random<sup>3</sup> feature, RICA [4] feature and our DDSFL features were extracted with step size 3 for the first layer, and step size 6 for the second and the third layer. SIFT features [1] were extracted from 16x16 patches, the step size was 3. HOG2x2 features [55] were extracted based on cells of size 8x8 with stride of 1 cell. LBP features [14] were extracted from cells of size 8x8.

We randomly picked 4,000 raw pixel value image patches (2,000 for MIT Indoor and PASCAL VOC 2012) from each training image with different scales, and select 10% of them as training exemplars (refer to Section 4). The margin of the hinge loss function in the objective function Equation 4 was fixed as 1. In each DDSFL layer, the parameters  $\xi$ ,  $\lambda_1$ ,  $\lambda_2$ ,  $\gamma$  and  $\eta$  were initialized as zeros, and they were sequentially learned one after another by applying cross validation (e.g., learn the best  $\xi$  first, and then fix the value of  $\xi$  and search for the best value of  $\lambda_1$ ). The maximum number of iterations of updating  $W$  and  $\alpha_c$  in Algorithm 1 was set to 5.

To convert the local features to global image representation, we utilized the LLC framework [10] and SPM [11], the combination of which can generally lead to good classification results. LLC used locality-constrained linear coding to encode local features, and performed max-pooling and linear-SVM afterwards. While SPM utilized rough spatial structure information, and concatenate all the pooled features from different spatial pooling regions. For all the local features, the size of the codebook was fixed as 2,000, and each image was divided into 1x1, 2x2, and 4x4 spatial pooling regions. We have also tested on other coding schemes (e.g. vector quantization) and pooling strategies (e.g. average pooling), our DDSFL consistently outperforms traditional local features.

In all the experiments of DDSFL, if not specified, we report the results of three layer learning framework with NN based exemplar selection method.

## 5.2. Comparison with Other Features

We firstly compare our DDSFL feature with the popular features which have shown good performance on scene images classification. There are three types of features: 1) Hand-crafted features, which include SIFT[1], HoG2x2[2, 55, 58], LBP[14], GIST[15], and CENTRIST[56]; 2) The baseline features learned in unsupervised way without encoding discriminative or shareable information, which include feature extracted with Random weights<sup>3</sup>, and the RICA [4] feature; 3) State-of-the-art Caffe[12] feature.

<sup>3</sup>DDSFL feature extraction scheme without training procedure, the filter bank  $W$  for each layer is randomly generated

Methods	aero	bike	bird	boat	bt1	bus	car	cat	chair	cow
SIFT[1]	77.8	44.4	42.4	54.0	16.0	74.7	52.1	53.0	45.6	21.4
HOG2x2[55]	78.3	44.8	38.9	51.6	16.6	77.0	51.2	57.3	49.2	23.5
LBP[14]	78.7	40.8	36.6	53.5	16.2	75.8	46.2	55.6	45.4	20.8
Random <sup>3</sup>	63.0	15.4	18.8	26.7	10.4	44.6	30.2	28.7	26.3	9.4
RICA	76.1	37.2	39.1	49.6	13.8	70.5	46.2	51.2	41.4	15.6
DDSFL	77.7	42.5	45.4	53.3	24.0	72.2	50.6	54.2	47.4	26.0
Caffe[12]	90.7	67.9	79.9	77.0	32.7	86.0	59.7	81.6	51.4	56.1
SIFT+Caffe	92.3	72.3	83.0	<b>79.5</b>	38.1	<b>88.5</b>	65.8	84.9	59.1	61.5
DDSFL+Caffe	<b>92.7</b>	<b>75.4</b>	<b>85.1</b>	79.4	<b>42.3</b>	88.1	<b>68.5</b>	<b>87.1</b>	<b>62.9</b>	<b>65.8</b>

Methods	table	dog	horse	moto	pers	plant	sheep	sofa	train	tv	mAP
SIFT[1]	40.2	38.4	40.7	54.2	71.3	13.1	27.4	33.0	64.5	47.6	45.6
HOG2x2[55]	40.4	40.3	40.6	46.2	72.8	13.5	34.4	34.7	63.6	49.9	46.2
LBP[14]	38.4	39.6	37.4	43.4	69.8	12.8	37.8	29.3	59.6	42.1	44.0
Random <sup>3</sup>	15.8	22.8	13.7	22.0	50.7	7.3	15.1	15.8	30.0	22.9	24.5
RICA[4]	29.8	35.6	33.3	46.6	67.0	14.1	28.9	29.6	58.9	44.3	41.4
DDSFL	36.6	42.6	38.9	50.7	70.3	23.4	36.2	36.4	61.6	49.0	46.9
Caffe[12]	55.2	72.0	69.5	75.1	88.0	36.3	63.2	43.0	87.7	64.6	66.9
SIFT+Caffe	<b>61.6</b>	76.0	72.7	<b>79.0</b>	90.7	41.8	69.2	52.7	<b>88.9</b>	70.1	71.4
DDSFL+Caffe	60.6	<b>80.2</b>	<b>74.0</b>	76.4	<b>92.3</b>	<b>43.6</b>	<b>74.3</b>	<b>53.2</b>	88.4	<b>73.7</b>	<b>73.2</b>

Table 2: Class-wise classification comparison results between DDSFL and other features on PASCAL VOC 2012.

As shown in Table 1 and Table 2, compared with the hand-crafted features and baseline unsupervised learned features, our DDSFL consistently and significantly outperforms all of them. By comparing with the baseline Random<sup>3</sup> features, the significant performance improvements demonstrate that the learning procedures can help to capture data adaptive information, and help to do better recognition. While by comparing with RICA features, the accuracy increasing indicates the effectiveness of learning discriminative and shareable information from image level labels.

### 5.2.1. Complementary Effect with Caffe

The Caffe feature [12] shown in Table 1 is a global feature extracted from a deep ConvNets [3], which is pre-trained on ImageNet [13]. The ConvNets has 7 layers (5 convolution layers + 2 fully connected layers). When applying the pre-trained model to extract features in datasets other than ImageNet, the 6th layer feature generally leads to better results than the 7th layer feature<sup>4</sup> [12, 25]. Therefore, we used the 6th layer Caffe feature for evaluation. Although the Caffe feature is very powerful, it is not fair to directly compare the performance of Caffe and the other features. Firstly, we did not have the huge amount of training data in ImageNet. Secondly, the color information has not been utilized<sup>5</sup>. Thirdly, we focused on learning powerful local features rather than global feature representation, and we supposed these two frameworks should encode complementary information, which was proved to be correct in the last row of Table 1 and Table 2.

To reveal the complementary effect intuitively, we tested on MIT 67 Indoor and compare the class-wise performance. In total, there are 14 categories that DDSFL performed better than Caffe, and 53 categories that Caffe did better than DDSFL. In Figure 3, the first two rows show the categories that our DDSFL worked better than Caffe, and we show the testing images which were correctly classified by DDSFL, but wrongly classified by Caffe. The last two rows show the categories which Caffe outperformed DDSFL, and

<sup>4</sup>We also tested with the 7th layer features. We got 87.35%, 93.44%, and 58.27% on the three scene datasets, and got mean accuracy of 65.79% on PASCAL VOC 2012.

<sup>5</sup>We did a small experiment with single layer DDSFL on UIUC sports, the inputs were L\*a\*b values image patches, around 1% performance gain can be achieved. However, this performance improvement was achieved with the price of three times of computation power, so we did not test on all the datasets.

auditorium				
		DSFL: 66.7%	Caffe: 38.89%	
corridor				
		DSFL: 57.14%	Caffe: 47.62%	
bowling				
		DSFL: 75.00%	Caffe: 100.00%	
winecellar				
		DSFL: 23.81%	Caffe: 76.19%	

Figure 3: Comparison results on MIT Indoor. The first two rows show the two categories on which DDSFL works better than Caffe, the last two rows show the classes that are better represented by Caffe. DDSFL and Caffe are complimentary.

we show the testing images which our DDSFL failed to recognize but Caffe could. To quantitatively analyze the complementation effect, we simply concatenate our DDSFL feature (after processing LLC and SPM) with the Caffe feature to check the performance. As shown in the last row of Table 1, we were able to get a huge performance improvement than purely using the Caffe features and produced the state-of-the-art results. ConvNets apply backpropagation to transmit the supervised information to all the deep layers, this is very helpful for the relatively high layer parameter updating, while it is extremely weak in training bottom layer parameters. In contrast, we explicitly use supervised information to train each layer. Moreover, our method is more suitable for relatively small datasets, as evidenced by the experimental results, while previous attempts on training a CNN classifier directly on small datasets usually failed. Thus, these two works are expected to be complimentary.

To make a fair comparison and exclude the influence brought by coding and pooling schemes, we also tested the combination of SIFT and Caffe. Although this combination also leads to satisfactory results, there is an obvious gap between the performance of SIFT+Caffe and DDSFL+Caffe. This observation indicates that our DDSFL can learn more effective complementary information by encoding data adaptive information. This might because of hand-crafted features such as SIFT usually extracting ‘garbor-like’ features, which are highly similar to the lower level filters learned by ConvNets and not as effectively complementary as our DDSFL.

### 5.3. Comparison with Other Methods on Scene Classification

There are numerous methods applied on the scene image classification, most of them simply utilized the existing hand-crafted features, and focus on the dictionary learning, coding scheme, or mid-level elements mining areas. Most of these works do not conflict with our feature learning framework, our DDSFL feature can easily replace the hand-crafted features, and combine with these methods to achieve better performance. (e.g. LScSPM [64] and IFV [39] focused on coding, which can be used to encode our DDSFL features.)

Methods	Scene 15	UIUC Sports	MIT Indoor
ROI + GIST[54]	-	-	26.50%
DPM [59]	-	-	30.40%
Object Bank [60]	80.90%	76.30%	37.60%
Discriminative Patches[50]	-	-	38.10%
LDC [49]	80.30%	-	43.53%
macrofeatures [61]	84.30%	-	-
Visual Concepts [36]	83.40%	84.80%	46.40%
SR-LSR [62]	85.70%	83.90%	-
MMDL [63]	86.35%	88.47%	50.15%
Discriminative Part Detector[38]	86.00%	86.40%	51.40%
LScSPM [64]	89.78%	85.27%	-
IFV [39]	-	-	60.77%
MLrep + IFV [37]	-	-	66.87%
ISPR + IFV [65]	<b>91.06%</b>	<b>92.08%</b>	68.50%
MOP-CNN (Caffe) [66]	-	-	<b>68.88%</b>
DDSFL + Caffe	<b>92.81%</b>	<b>96.78%</b>	<b>76.23%</b>

Table 3: Comparison results of our method and other popular methods on Scene 15, UIUC sports, and MIT Indoor.

As shown in Table 3, comparing with the other methods, our DDSFL+Caffe feature achieved the highest accuracy results on all of the three scene classification datasets. Note that Visual Elements [37] utilized numerous patches extracted at scales ranging from 80x80 to the full image size, and the patches were represented by standard HOG [2] plus a 8x8 color image in L\*a\*b space, and very high dimensional IFV[39] features. While MMDL [63] combined 5 types of features on 3 scales. MOP-CNN (Caffe) [66]) enhanced the original Caffe with 10% by considering geometric invariance, which can also be used in our method to further improve the performance.

#### 5.4. Analysis of The Effects of Different Components

In this section, we aim to compare our shareable and discriminative learning method to the baseline without encoding such information, which is equivalent to the RICA method in [4], and the baseline of using randomly generated feature transformation matrix without learning procedure at all. In the rest of this subsection, we report the results of recruiting different components of the 2 layer DDSFL, which can be efficiently tested and already contain the hierarchical information.

We first show the visualization of the first layer filters learned from UIUC Sports in Figure 4(a) and Figure 4(b). We can see that our DDSFL is able to capture more sharply localized patterns, corresponding to more class-specific visual information.

##### 5.4.1. Effect of Learning Shareable Filters

We tested the DDSFL with or without the feature sharing terms, and got the intermediate results in Table 4. The first row of the table shows the result of plainly using randomly generated filters without learning, the poor results indicate that the representation power of the network structure itself is severely limited. The second row shows the unsupervised RICA features learned by solving Equation 1. In the third row,  $L_u + L_{sha}$  corresponds to features learned with Equation 2. The improvement in accuracy shows that learning shareable features is effective for classification. However, if we removed the global reconstruction error term  $\mathcal{L}_u$  and only kept the shareable terms, as shown in the fourth row, the performance dramatically dropped to a random filter bank level.

##### 5.4.2. Effect of Discriminative Regularization and Exemplar Selection

Comparing the fifth to seventh rows of Table 4, we can observe that without applying exemplars selection step, we could not achieve much improvement because noisy training examples might overwhelm the useful



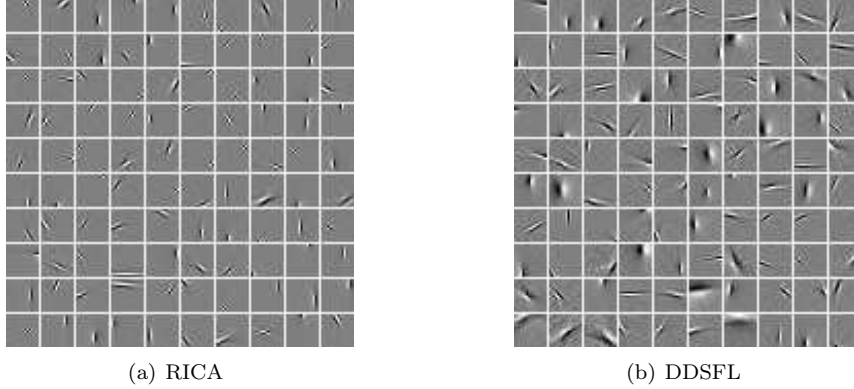


Figure 4: Visualization of the filters learned by RICA and DDSFL on the UIUC Sports.

Methods	Scene 15	UIUC Sports	MIT Indoor
Random <sup>3</sup>	73.52%	70.42%	23.28%
$L_u$ (RICA[4])	79.85%	82.14%	47.89%
$L_u + L_{sha}$	82.01%	83.67%	49.70%
$L_{sha}$	72.69%	72.52%	24.12%
$L_u + L_{sha} + L_{dis}$ (without Exemplar)	82.50%	83.43%	51.28%
$L_u + L_{sha} + L_{dis}$ (SVM Exemplar)	83.02%	84.98%	51.65%
$L_u + L_{sha} + L_{dis}$ (NN Exemplar)	<b>84.19%</b>	<b>86.45%</b>	<b>52.24%</b>

Table 4: Analysis of the effect of each component in DDSFL (two layers)

discriminative patterns. However, once using the selected exemplars, our method could achieve significant improvement in classification accuracy. This shows that discriminative exemplar selection is critical in our learning framework. Furthermore, as shown in the last two rows of Table 4 comparing the NN based and SVM based exemplar selection methods, the former method usually got better performance with higher computational and memory cost.

Furthermore, it is obvious that only using 10% of the whole patch set dramatically increased the efficiency of nearest neighbor search in the training step. Thus, our exemplar selection method is both effective and efficient.

#### 5.4.3. Effect of Hierarchical Structure

According to the comparison results shown in Table 5, we can observe three interesting phenomena of using different number of layers. Firstly, no matter using how many layers, learning data adaptive features can always achieve more powerful filter banks and lead to better recognition results, and considering shareable and discriminative information can consistently achieve better result than unsupervised learning. Secondly, the good performance of higher layer features heavily depends the high quality lower level features, by using the Random filters, the more layers be processed, the worse results will be achieved. Thirdly, in the RICA and our DDSFL cases, we observe that the performance gain of using two layer structure is much more obvious than adding the third layer, which might because of lacking of training samples to support lager learning framework (especially when the scale is very small, one image might contain only one 3rd layer patch).

#### 5.4.4. Effect of The Size of Filter Bank

To further analyze the influence caused by the size of filters, we tested on Scene 15 dataset with 128, 256, 512, 1024, and 2048 filters by using different number of learning layers. The results are shown in Figure 5. The accuracy results consistently increase when the number of filters increase. When the size of

Methods	Scene 15	UIUC Sports	MIT Indoor
Random <sup>3</sup> (1 layer)	74.12%	71.46%	23.53%
Random <sup>3</sup> (2 layers)	73.52%	70.42%	23.28%
Random <sup>3</sup> (3 layers)	70.91%	67.92%	21.34%
RICA[4] (1 layer)	77.54%	79.03%	41.36%
RICA[4] (2 layers)	79.85%	82.14%	47.89%
RICA[4] (3 layers)	80.01%	82.45%	47.76%
DDSFL (1 layer)	82.61%	83.92%	47.16%
DDSFL (2 layers)	84.19%	86.45%	52.24%
DDSFL (3 layers)	<b>84.42%</b>	<b>86.91%</b>	<b>52.26%</b>

Table 5: Analysis of the effect of number of layers

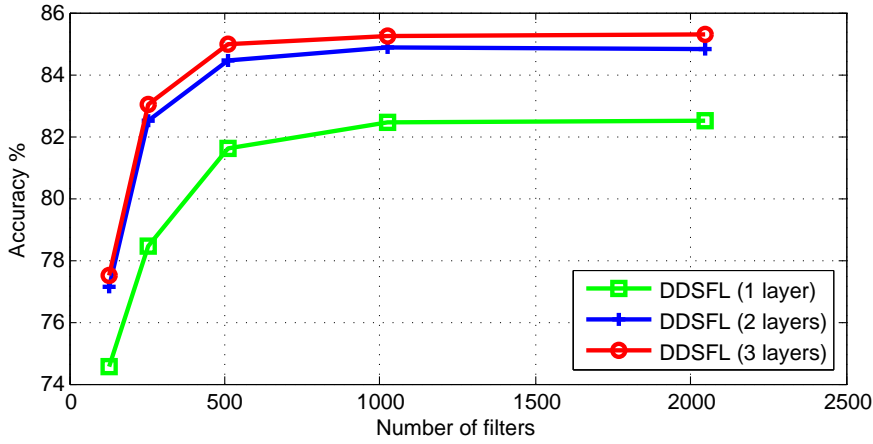


Figure 5: Results of varying number of filters and layers in Scene 15

the filter bank is small, especially when the feature transformation matrix is under-complete, the learned features are relatively weak. When the number of filters increases, and  $W$  becomes over-complete, the performance is substantially improved. Thus, learning over-complete filter banks do help to obtain better feature representation because the resulting filter banks captures more information. However, when the number of filters further increases, the performance does not increase much, while the learning process will be extremely slow. In our experiment, we learned 400 filters for all the layers as a compromise between efficiency and accuracy. Furthermore, with different number of filters, increasing the number of DDSFL layers also consistently increases the performance, which indicates that higher level DDSFL features can learn complementary patterns.

## 6. Conclusion

In this paper, we propose a hierarchical weakly supervised local feature learning method, called DDSFL, to learn discriminative and shareable filter banks to transform local image patches into different visual level features. In our DDSFL method, we learn a flexible number of shared filters to represent shareable patterns exist among similar categories. To enhance the discriminative power, we force the features from the same class to be similar, while features from different classes to be separable. We tested our method on three scene image classification benchmark datasets and PASCAL VOC 2012, the results consistently show that our learned features can outperform most of the existing features. By combining with the Caffe features pre-trained on ImageNet, we can greatly enhance the representation power, and achieve state-of-the-art classification results on all the three scene datasets, and also get promising results on PASCAL.

## Acknowledgment

The research is supported by Singapore Ministry of Education (MOE) Tier 2 ARC28/14, and Singapore A\*STAR Science and Engineering Research Council PSF1321202099.

## References

- [1] D. G. Lowe, Distinctive image features from scale-invariant keypoints, *International journal of computer vision* 60 (2) (2004) 91–110.
- [2] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: *CVPR*, 2005.
- [3] A. Krizhevsky, I. Sutskever, G. Hinton, Imagenet classification with deep convolutional neural networks, in: *NIPS*, 2012.
- [4] Q. V. Le, A. Karpenko, J. Ngiam, A. Y. Ng, Ica with reconstruction cost for efficient overcomplete feature learning, in: *NIPS*, 2011.
- [5] W. Y. Zou, S. Y. Zhu, A. Y. Ng, K. Yu, Deep learning of invariant features via simulated fixations in video, in: *NIPS*, 2012.
- [6] G. Hinton, S. Osindero, Y.-W. Teh, A fast learning algorithm for deep belief nets, *Neural computation* 18 (7) (2006) 1527–1554.
- [7] A. Coates, H. Lee, A. Y. Ng, An analysis of single-layer networks in unsupervised feature learning, in: *AI Statistics*, 2011.
- [8] K. Sohn, D. Y. Jung, H. Lee, A. O. Hero, Efficient learning of sparse, distributed, convolutional feature representations for object recognition, in: *ICCV*, 2011.
- [9] Z. Zuo, G. Wang, Learning discriminative hierarchical features for object recognition, *Signal Processing Letters* 21 (9) (2014) 1159–1163.
- [10] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, Y. Gong, Locality-constrained linear coding for image classification, in: *CVPR*, 2010.
- [11] S. Lazebnik, C. Schmid, J. Ponce, Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, in: *CVPR*, 2006.
- [12] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, T. Darrell, Decaf: A deep convolutional activation feature for generic visual recognition, in: *ICML*, 2014.
- [13] J. Deng, A. C. Berg, K. Li, L. Fei-Fei, What does classifying more than 10,000 image categories tell us?, in: *ECCV*, 2010.
- [14] T. Ojala, M. Pietikainen, T. Maenpaa, Multiresolution gray-scale and rotation invariant texture classification with local binary patterns, *IEEE Trans. Pattern Analysis and Machine Intelligence* 24 (7) (2002) 971–987.
- [15] A. Oliva, A. Torralba, Building the gist of a scene: The role of global image features in recognition, *Progress in brain research* 155 (2006) 23–36.
- [16] Q. V. Le, M. A. Ranzato, R. Monga, M. Devin, K. Chen, G. S. Corrado, J. Dean, A. Y. Ng, Building high-level features using large scale unsupervised learning, in: *ICML*, 2012.
- [17] Q. V. Le, W. Y. Zou, S. Y. Yeung, A. Y. Ng, Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis, in: *CVPR*, 2011.
- [18] A. Wang, J. Lu, G. Wang, J. Cai, T.-J. Cham, Multi-modal unsupervised feature learning for rgb-d scene labeling, in: *ECCV*, 2014.
- [19] D. Wang, X. Tan, C-svddnet: An effective single-layer network for unsupervised feature learning, *arXiv preprint arXiv:1412.7259*.
- [20] J. Yang, J. Gao, G. Wang, S. Zhang, Natural scene recognition based on superpixels and deep boltzmann machines, *arXiv preprint arXiv:1506.07271*.
- [21] J. Gao, J. Yang, G. Wang, M. Li, A novel feature extraction method for scene recognition based on centered convolutional restricted boltzmann machines, *arXiv preprint arXiv:1506.07257*.
- [22] A. Dosovitskiy, P. Fischer, J. T. Springenberg, M. Riedmiller, T. Brox, Discriminative unsupervised feature learning with exemplar convolutional neural networks, *arXiv preprint arXiv:1506.02753*.
- [23] S. Bianco, G. Ciocca, C. Cusano, Curl: Co-trained unsupervised representation learning for image classification, *arXiv preprint arXiv:1406.6909*.
- [24] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, Y. LeCun, Overfeat: Integrated recognition, localization and detection using convolutional networks, *arXiv preprint arXiv:1312.6229*.
- [25] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, *arXiv preprint arXiv:1311.2524*.
- [26] Z. Zuo, B. Shuai, G. Wang, X. Liu, X. Wang, B. Wang, Y. Chen, Convolutional recurrent neural networks: Learning spatial dependencies for image representation, in: *CVPRW*, 2015.
- [27] S. H. Khan, M. Hayat, M. Bennamoun, R. Togneri, F. Sohel, A discriminative representation of convolutional features for indoor scene recognition, *arXiv preprint arXiv:1506.05196*.
- [28] M. Hayat, S. H. Khan, M. Bennamoun, S. An, A spatial layout and scale invariant feature representation for indoor scene classification, *arXiv preprint arXiv:1506.05532*.
- [29] B. Shuai, G. Wang, Z. Zuo, B. Wang, L. Zhao, Integrating parametric and non-parametric models for scene labeling, in: *CVPR*, 2015.
- [30] W. Shen, X. Wang, Y. Wang, X. Bai, Z. Zhang, Deepcontour: A deep convolutional feature learned by positive-sharing loss for contour detection, in: *CVPR*, 2015.
- [31] Y. Wang, S. Li, A. C. Kot, Complementary feature extraction for branded handbag recognition, in: *ICIP*, 2014.

- [32] Z. Jiang, Z. Lin, L. S. Davis, Learning a discriminative dictionary for sparse coding via label consistent k-svd, in: CVPR, 2011.
- [33] J. Mairal, F. Bach, J. Ponce, G. Sapiro, A. Zisserman, Supervised dictionary learning, in: NIPS, 2008.
- [34] M. Yang, L. Zhang, X. Feng, D. Zhang, Fisher discrimination dictionary learning for sparse representation, in: ICCV, 2011.
- [35] S. Kong, D. Wang, A dictionary learning approach for classification: separating the particularity and the commonality, in: ECCV, 2012.
- [36] Q. Li, J. Wu, Z. Tu, Harvesting mid-level visual concepts from large-scale internet images, in: CVPR, 2013.
- [37] C. Doersch, A. Gupta, A. A. Efros, Mid-level visual element discovery as discriminative mode seeking, in: NIPS, 2013.
- [38] J. Sun, J. Ponce, et al., Learning discriminative part detectors for image classification and cosegmentation, in: ICCV, 2013.
- [39] M. Juneja, A. Vedaldi, C. Jawahar, A. Zisserman, Blocks that shout: Distinctive parts for scene classification, in: CVPR, 2013.
- [40] B. Yao, L. Fei-Fei, Action recognition with exemplar based 2.5 d graph matching, in: ECCV, 2012.
- [41] T. Malisiewicz, A. Gupta, A. A. Efros, Ensemble of exemplar-svms for object detection and beyond, in: ICCV, 2011.
- [42] C. Li, Q. Liu, W. Dong, X. Zhang, L. Yang, Max-margin based discriminative feature learning, arXiv preprint arXiv:1412.4863.
- [43] H. O. Song, S. Zickler, T. Althoff, R. Girshick, M. Fritz, C. Geyer, P. Felzenszwalb, T. Darrell, Sparselet models for efficient multiclass object detection, in: ECCV, 2012.
- [44] H. O. Song, T. Darrell, R. B. Girshick, Discriminatively activated sparselets, in: ICML, 2013.
- [45] Z. Zuo, G. Wang, B. Shuai, L. Zhao, Q. Yang, X. Jiang, Learning discriminative and shareable features for scene classification, in: ECCV, 2014.
- [46] G. E. Hinton, R. R. Salakhutdinov, Reducing the dimensionality of data with neural networks, *Science* 313 (5786) (2006) 504–507.
- [47] O. Boiman, E. Shechtman, M. Irani, In defense of nearest-neighbor based image classification, in: CVPR, 2008.
- [48] S. McCann, D. G. Lowe, Local naive bayes nearest neighbor for image classification, in: CVPR, 2012.
- [49] Z. Wang, J. Feng, S. Yan, H. Xi, Linear distance coding for image classification, *IEEE Trans. Image Processing* 22 (2) (2013) 537–548.
- [50] S. Singh, A. Gupta, A. A. Efros, Unsupervised discovery of mid-level discriminative patches, in: ECCV, 2012.
- [51] M. Muja, D. G. Lowe, Fast approximate nearest neighbors with automatic algorithm configuration, in: VISSAPP, 2009.
- [52] G. Wang, D. Hoiem, D. Forsyth, Learning image similarity from flickr groups using fast kernel machines, *IEEE Trans. Pattern Analysis and Machine Intelligence* 34 (11) (2012) 2177–2188.
- [53] L.-J. Li, L. Fei-Fei, What, where and who? classifying events by scene and object recognition, in: ICCV, 2007.
- [54] A. Quattoni, A. Torralba, Recognizing indoor scenes, in: CVPR, 2009.
- [55] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, A. Torralba, Sun database: Large-scale scene recognition from abbey to zoo, in: CVPR, 2010.
- [56] J. Wu, J. M. Rehg, Centrist: A visual descriptor for scene categorization, *IEEE Trans. Pattern Analysis and Machine Intelligence* 33 (8) (2011) 1489–1501.
- [57] R. Margolin, L. Zelnik-Manor, A. Tal, Otc: A novel local descriptor for scene classification, in: ECCV, 2014.
- [58] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, D. Ramanan, Object detection with discriminatively trained part-based models, *IEEE Trans. Pattern Analysis and Machine Intelligence* 32 (9) (2010) 1627–1645.
- [59] M. Pandey, S. Lazebnik, Scene recognition and weakly supervised object localization with deformable part-based models, in: ICCV, 2011.
- [60] L.-J. Li, H. Su, L. Fei-Fei, E. P. Xing, Object bank: A high-level image representation for scene classification & semantic feature sparsification, in: NIPS, 2010.
- [61] Y. L. Boureau, Y. L. F. Bach, J. Ponce, Learning mid-level features for recognition, in: CVPR, 2010.
- [62] X. Li, Y. Guo, Latent semantic representation learning for scene classification, in: ICML, 2014.
- [63] X. Wang, B. Wang, X. Bai, W. Liu, Z. Tu, Max-margin multiple-instance dictionary learning, in: ICML, 2013.
- [64] S. Gao, I.-H. Tsang, L.-T. Chia, Laplacian sparse coding, hypergraph laplacian sparse coding, and applications, *IEEE Trans. Pattern Analysis and Machine Intelligence* 35 (1) (2013) 92–104.
- [65] D. Lin, C. Lu, R. Liao, J. Jia, Learning important spatial pooling regions for scene classification, in: CVPR, 2014.
- [66] Y. Gong, L. Wang, R. Guo, S. Lazebnik, Multi-scale orderless pooling of deep convolutional activation features, in: ECCV, 2014.