Hindawi
Mathematical Problems in Engineering
Volume 2017, Article ID 5218247, 16 pages
https://doi.org/10.1155/2017/5218247

Research Article

Image Classification Based on Convolutional Denoising Sparse Autoencoder

Shuangshuang Chen,[1,2] Huiyi Liu,[1] Xiaoqin Zeng,[1] Subin Qian,[1,2] Jianjiang Yu,[2] and Wei Guo[2]

[1] *Institute of Intelligence Science and Technology, Hohai University, No. 8 West Focheng Road, Nanjing 211100, China*
[2] *School of Information Science and Technology, Yancheng Teachers University, Yancheng 224002, China*

Correspondence should be addressed to Shuangshuang Chen; chenss@yctu.edu.cn

Received 15 March 2017; Accepted 16 September 2017; Published 26 November 2017

Academic Editor: Lotfi Senhadji

Image classification aims to group images into corresponding semantic categories. Due to the difficulties of interclass similarity and intraclass variability, it is a challenging issue in computer vision. In this paper, an unsupervised feature learning approach called convolutional denoising sparse autoencoder (CDSAE) is proposed based on the theory of visual attention mechanism and deep learning methods. Firstly, saliency detection method is utilized to get training samples for unsupervised feature learning. Next, these samples are sent to the denoising sparse autoencoder (DSAE), followed by convolutional layer and local contrast normalization layer. Generally, prior in a specific task is helpful for the task solution. Therefore, a new pooling strategy—spatial pyramid pooling (SPP) fused with center-bias prior—is introduced into our approach. Experimental results on the common two image datasets (STL-10 and CIFAR-10) demonstrate that our approach is effective in image classification. They also demonstrate that none of these three components: local contrast normalization, SPP fused with center-prior, and l_2 vector normalization can be excluded from our proposed approach. They jointly improve image representation and classification performance.

1. Introduction

In recent years, image classification has been an active and important research topic in the field of computer vision and machine learning applications. The basic image classification algorithm is generally introduced in [1–3] and involves three main stages in sequence: (1) image sampling, (2) feature extraction, and (3) classifier designing. In these stages, feature extraction plays an important role [4], and the efficient features extracted may increase the separation between spectrally similar classes, resulting in improved classification performance.

The feature extraction part is commonly accomplished by a wide spectrum of different local or global descriptors, for example, scale invariant feature transform (SIFT) [5], histogram of oriented gradients (HOG) [6], and local binary pattern (LBP) [7]. Although these hand-crafted features lead to reasonable results in various applications, they are only suitable for a particular data type or research domain and would result in dismal performance on other unknown usage [2]. Recently, there is a growing consensus that it is an alternative approach to utilize deep learning methods to obtain machine-learned features for image classification. These deep learning methods aim to extract general purpose features for any images rather than learning domain adaptive feature descriptors particularly for certain tasks.

Up to this point, typical deep learning methods include convolutional neural network (CNN) [8, 9], sparse coding [10], deep belief network (DBN) [11] and stacked autoencoder (AE) [12]. Among these models, CNN is one of the main models in deep learning methods, which is a hierarchal model that outperforms many algorithms on visual recognition tasks. One property is alternately using convolution [13] and pooling [14] structures. The convolution operation shares weights and keeps the relative location of features and thus can preserve spatial information of the input data. Despite its apparent success, there remains a major drawback: CNN requires large quantities of labeled data, which are very

expensive to obtain. Stacked AE is another notable learning method, which exploits a particular type of neural network: the AE, also called autoassociator [15]—as component or monitoring device. It can be effectively used for unsupervised feature learning on a dataset for which it is difficult to obtain labeled samples [16]. Beyond simply learning features by AE, there is a need for reinforcing the sparsity of weights and increasing its robustness to noise. Ng [17] introduced sparse autoencoder (SAE), which is a variant of AE. Sparsity is a useful constraint when the number of hidden units is large. SAE has very few neurons that are active. There is an another variant of AE called denoising autoencoder (DAE) [18], which minimizes the error in reconstructing the input from a stochastically corrupted transformation of the input. The stochastic corruption process consists in randomly setting some of inputs (as many as half of them) to zero. Comparative experiments clearly show the surprising advantage of DAE on a pattern classification benchmark suite.

With the development of deep learning, AE and its variants are widely used in the field of image recognition. Xu et al. [19] presented a stacked SAE for nuclei patch classification on breast cancer histopathology. They extracted two classes of 34×34 patches from the histopathology images: nuclei and nonnuclei patches. These two kinds of patches were used to construct the training set and testing set. The authors of [20] proposed a method called stacked DAE based on paper [18], which is a straightforward variation on the stacked ordinary AE. Besides, stacked DAE was tested on MINIST dataset, which contains 28×28 gray-scale images. Being similar to the method based on stacked SAE, the training and testing dataset fed into models are relatively low in resolution, such as small image patches and low resolution images (e.g., hand-written digits). Both SAE and DAE are common fully connected networks, which cannot scale well to realistically sized high-dimensional inputs (e.g., 256×256 images) in terms of computational complexity [21]. They both ignore the 2D image structure.

In order to overcome these limitations, this paper introduces an approach called CDSAE (convolutional denoising sparse autoencoder) that scales well to high-dimensional inputs. This approach can effectively integrate the advantages in SAE, DAE, and CNN. This hybrid structure forces our model to learn more abstract and noise-resistant features, which will help to improve the model's representation learning performance. CDSAE can map images to feature representation without any label information, while CNN requires large quantities of labeled data. Besides, it differs from conventional SAE and DAE as its weights are shared among all locations in the input images and thus preserves spatial locality.

Besides feature extraction mentioned above, the sampler is another critical component which has a great influence on the results. Ideally, it should focus attention on the image regions that are the most informative for classification [22]. Recently, selective attention models have drawn a lot of research attention [23, 24]. The idea in selective attention is that not all parts of an image give us information. If we can attend only to the relevant parts, we can recognize the image more quickly and using less resources [23]. People place an object on the foveal with fixations when the gaze is concentrated on the object and get most information through fixations [25]. Compared to the traditional approaches using a random sampling strategy, we introduce a sampling strategy to sample fixations from the image, which is inspired by human selective attention. Moreover, those studies on human eye fixations demonstrate that there is a tendency in humans to look towards the image center, which is called the center bias [26]. It is worth mentioning that incorporating center-bias prior into saliency estimation has been previously investigated by a number of researchers [27–29]. Turning to our work, center-bias prior are absorbed for SPP in our image classification model.

To summarize, the key contributions of this paper are elaborated as follows:

(1) A sampling strategy about eye fixations based on human visual system is proposed, which is inspired by human eyes. The fixation points and nonfixation points of images can be got by utilizing saliency detection model.

(2) A CDSAE model with local contrast normalization operation is proposed. In this overall model, single-layer DSAE is used for unsupervised feature learning, which can effectively extract features without using any label data. Compared to conventional deep models, single-layer DSAE has a strength with a smaller computational learning cost and fewer hyperparameters to tune.

(3) An SPP incorporating center-bias prior is proposed. This not only maintains spatial information by pooling in local spatial bins but also fully utilizes prior knowledge of image dataset. To the best of our knowledge, this is the first work that absorbs prior knowledge for pooling in image classification.

The remainder of this paper is organized as follows. In Section 2, we review related works in the literature. Section 3 introduces a sampling strategy based on human vision attention system. Section 4 describes CDSAE and Section 5 provides the overall classification framework. The details of our experiments and the results are presented in Section 6, followed by a discussion and future work.

2. Related Work

Other researchers have also made some headway on constructing the convolutional autoencoder (CAE), an unsupervised feature extractor that can scale well to high-dimensional input images. Masci et al. [21] propose a kind of CAE, which directly takes the high-dimensional image data as the input through training the AE convolutionally. Though this convolution structure can preserve local relevance of the inputs, training the AE convolutionally is not easy. For this problem, Coates et al. [30] first extract patches from the input images and use patch-wise training to optimize the weights of a basic SAE in place of convolutional training. Besides, they further propose that, even with a single-layer network in unsupervised feature learning, it is possible to achieve

state-of-the-art performance. In our method, we absorb this idea and construct a single-layer network for unsupervised feature learning. Due to its simplicity and efficiency, single-layer SAE has a wide range of applications. Luo et al. [3] utilize single-layer SAE for natural scene classification; this idea is analogous to Coates et al.'s work [30]. Similar method is used for remote sensing image classification reported in [31]. These locally connected SAE through convolution [3, 30, 31] present many similarities with each layer of CNN, such as the use of convolution and pooling.

There are several differences between these works and ours. Firstly, we adopt the theory of DAE, which can learn more noise-resistant features. Hence, our model is more significant unlike previous works which only use sparsity. Secondly, local contrast normalization layer is embedded before pooling layer in our model. In [32], He et al. introduce a spatial pyramid pooling (SPP), which shows great strength in object detection. In contrast to [3, 30, 31] which only use single-level pooling, we instead propose an SPP fused with center-bias prior. Bias is mainly proposed for image saliency detection in computer vision. It is often closely related to the application task and could be deliberately utilized as a prior in specific task to improve the performance of the task [33].

Another branch of related works are human selective attention models. Many attention models have been proposed in both natural language processing and computer vision. In [34], Wang et al. have proven that human read sentences by making a sequence of fixations and saccades. They explore attention models over single sentences with guidance of human attention. In computer vision area, the core concept of attention models is to focus on the important parts of the input image, instead of giving all pixels the same weight [34]. Inspired by the theory of visual attention mechanism, we propose a sampling strategy about eye fixations based on human visual system. Our work is also closely related to the work of Judd et al. [35] who train a model of saliency directly from human fixations data. Saliency map computed by saliency detection models is significantly correlated with human fixation patterns [36].

3. Sampling Strategy Based on Human Vision Attention System

Methods of saliency detection proposed are selective attention models which simulate visual attention system. They can be used to measure the conspicuity of a location, or the likelihood of a location to attract the attention of human observers [35]. The saliency map represents the saliency of each pixel. And it can be thresholded such that a given percent of the image pixels are classified as fixated and the rest are classified as not fixated [35]. In this paper, we adopt a saliency detection model—context-aware saliency—to guide our sampling task [37]. It is a new type of saliency detection algorithm which manages to detect the pixels on the salient objects and only them. This method has proposed that a pixel i is considered salient if the appearance of the patch p_i centered at pixel i is distinctive with respect to all other image patches. $d_{\text{color}}(p_i, p_j)$ is the Euclidean distance between the patches p_i and p_j in the CIE $L * a * b$ color space, normalized to the range $[0, 1]$. If $d_{\text{color}}(p_i, p_j)$ is high $\forall j$, then pixel i is considered salient. And $d_{\text{position}}(p_i, p_j)$ denotes the Euclidean distance between the positions of patches p_i and p_j, normalized by the larger image dimension. A dissimilarity measure is defined between a pair of patches as

$$d\left(p_i, p_j\right) = \frac{d_{\text{color}}\left(p_i, p_j\right)}{1 + c \cdot d_{\text{position}}\left(p_i, p_j\right)}, \tag{1}$$

where $c = 3$ in our paper. This dissimilarity measure is proportional to the distance in color space and inversely proportional to the positional distance. For every patch p_i, we search for the L most similar patches $\{q_j\}_{j=1}^{L}$ in the image (if the most similar patches are highly different from p_i, then clearly all image patches are highly different from p_i). As stated before, a pixel i is salient when $d(p_i, p_j)$ is high $\forall j \in [1, L]$. Therefore, the single-scale saliency value of pixel i at scale r can be defined as

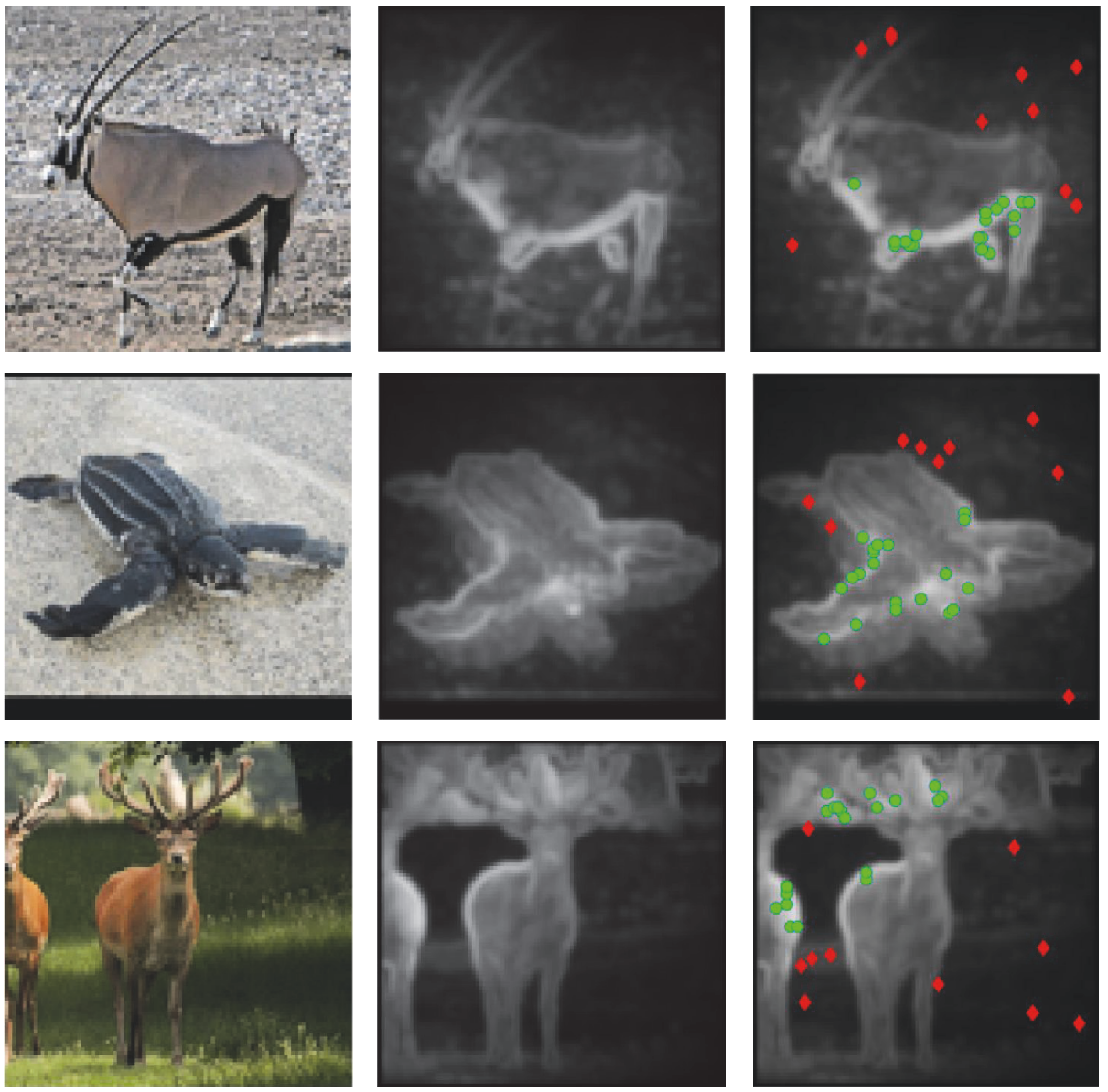
$$S_i^r = 1 - \exp\left\{-\frac{1}{L}\sum_{l=1}^{L} d\left(p_i^r, q_l^r\right)\right\}. \tag{2}$$

Furthermore, we also use four scales (100%, 80%, 50%, and 30%) of the original image to measure the saliency in a multiscale image. The saliency at pixel i is taken as the mean of its saliency at different scales (more details can be found in [37]).

60% of the ground truth human fixations are within the top 5% salient areas of a saliency map, and 90% are within the top 20% salient locations [35]. Saliency map can be thresholded such that a given percent of the image pixels are classified as fixation and the rest are classified as nonfixations. Figure 1 shows the saliency detection results for three images of STL-10 dataset. To avoid missing the nonfixations corresponding to the images, we also sample some nonfixations. Figure 1(c) shows the fixations and the nonfixations in each image. Thus, we first randomly select one image of M images and then extract a given percent of the fixations and nonfixations. For each image, the total number of the fixations and nonfixations is N. This can be represented as a vector in $\mathbb{R}^C$ of the pixel intensity values, with $C = N \times 3$ (the input image has three channels—R, G, and B). Therefore, a dataset $X \in \mathbb{R}^{C \times M}$ can thus be constructed, where each column denotes the pixel intensity values of the fixations and nonfixations sampled from each image.

4. Convolutional Denoising Sparse Autoencoder

CDSAE can be divided into three stages: feature learning, feature extraction, and classification. These stages are in correspondence with (1) training the single-layer DSAE; (2) convolution, local contrast normalization, and SPP fused with center-bias prior; (3) support vector machine (SVM) classification. The power of DSAE lies in the form of reconstruction-oriented training, where the hidden units can conserve the efficient feature to represent the input data. In order to get better representation, the convolution operation



(a) (b) (c)

Figure 1: (a) Sample images from STL-10 dataset. (b) Saliency maps for original images. (c) Several human fixations and nonfixations of images. (The green points of circle denote fixations and the red points of diamond denote nonfixations.)

is introduced to encode the input images with the features extracted by DSAE. A local contrast normalization layer is embedded after convolution operation. This can improve feature invariance and increases sparsity (Large-Scale Visual Recognition with Deep Learning http://cvgl.stanford.edu/teaching/cs231a_winter1314/lectures/lecture_guest_ranzato.pdf). Following the local contrast normalization, pooling is conducted to select significant features and decreases the spatial resolution. Several types of pooling method have been proposed to subsample the features, for example, average pooling [46], max pooling [47], stochastic pooling [44], and spatial pyramid pooling [32]. We propose a new form of SPP which seamlessly incorporate center-bias prior.

Figure 2 shows how the CDSAE works.

4.1. Feature Learning. Recently, increasing attention has been drawn to the study of single-layer network for unsupervised feature learning [3, 31]. Paper [30] has proved that simple but fast algorithms can be highly competitive, while more complex algorithms may have greater complexity and expense. In order to extract appropriate and sufficient features with low computational cost, a single-layer DSAE model is proposed in this work. The DSAE is a simple but effective extension of the classical SAE. The main idea of this approach is to train a sparse AE which could reconstruct the input data from a corrupted version by manual addition with random noise.

4.1.1. Sparse Autoencoder. AE is a symmetrical neural network structurally defined by three layers: input layer, hidden

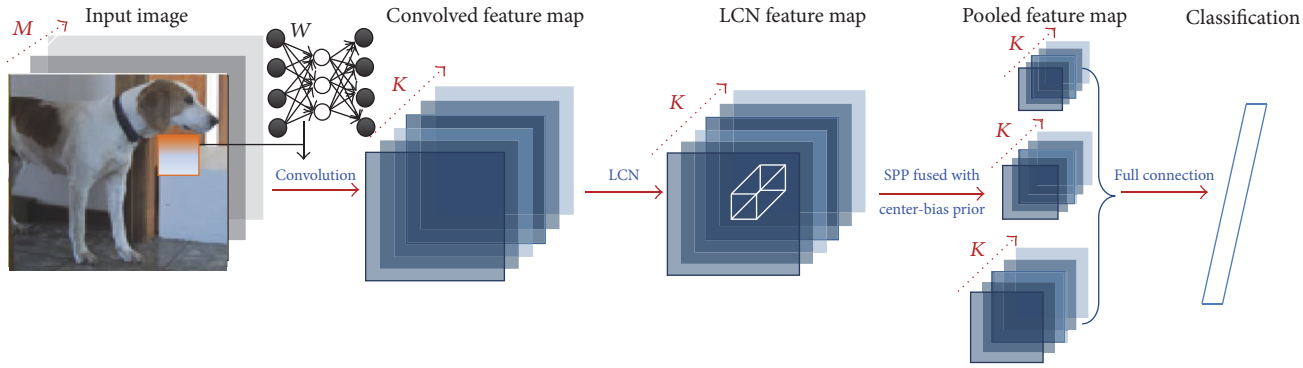


Figure 2: The flowchart of the CDSAE.

layer, and output layer. It can be used to learn the features of a dataset in an unsupervised manner. The aim of the AE is to learn a latent or compressed representation of the input data, by minimizing the reconstruction error between the input at the encoding layer and its reconstruction at the decoding layer.

During the encoding step, an input vector $x^i \in \mathbb{R}^C$ is processed by applying a linear deterministic mapping and a nonlinear activation function l as follows:

$$\alpha^i = f\left(x^i\right) = l\left(W_1 x^i + b_1\right), \tag{3}$$

where $W_1 \in \mathbb{R}^{K\times C}$ is a weight matrix with K features and $b_1 \in \mathbb{R}^K$ is the encoding bias. In this study, we consider a leaky rectified linear unit (LReLU) activation function for $l(x)$. Because LReLU has better performance than ReLU, it is widely used in the field of deep learning [48–50]. It can be represented as

$$y = \begin{cases} x & \text{if } x \geq 0 \\ \varpi x & \text{if } x \leq 0, \end{cases} \tag{4}$$

and the slope ϖ of the LReLU is set to 0.01 [48]. Then we decode a vector using a separate linear decoding matrix

$$z^i = W_2 \alpha^i + b_2, \tag{5}$$

where $W_2 \in \mathbb{R}^{C\times K}$ and $b_2 \in \mathbb{R}^C$ are a decoding weight matrix and a bias vector, respectively. Feature extractors are learned by minimizing the reconstruction error of the cost function in (6). The first term in the cost function is the error term. The second term is a regularization term (a.k.a. a weight decay term).

$$L(X, Z) = \frac{1}{2}\sum_{i=1}^{M}\left\|x^i - z^i\right\|^2 + \frac{\lambda}{2}\|W\|^2, \tag{6}$$

where X and Z represent the training and reconstructed data, respectively.

In order for the sparseness of hidden units, the method of [51] is introduced to constrain the expected activation of hidden nodes. We add a regularization term that penalizes the values of hidden units, such that only a few of them are bigger than the sparsity parameter ρ and most values of hidden units are much smaller than ρ. $\mathrm{KL}(\rho \parallel \widehat{\rho})$ is the sparse penalty term, which can be denoted as the following formula:

$$\mathrm{KL}\left(\rho \parallel \widehat{\rho}\right) = \rho \log \frac{\rho}{\widehat{\rho}} + (1-\rho)\log \frac{1-\rho}{1-\widehat{\rho}}, \tag{7}$$

where $\mathrm{KL}(\cdot)$ is the Kullback–Leibler divergence [52]. We recall that α denotes the activation of hidden units in autoencoder; let $\widehat{\rho} = (1/M)\sum_1^M[\alpha^{(i)}]$ be the average activation of α averaged over the training set $X^{C\times M}$. Then our objective function in the sparse autoencoder learning can be written as follows:

$$L(X, Z) + \beta\sum_{j=1}^{K}\mathrm{KL}\left(\rho \parallel \widehat{\rho}\right). \tag{8}$$

With the introduction of the KL divergence weighted by a sparsity penalty parameter β in the objective function, we penalize a large average activation of α over the training samples by setting ρ small. This penalization drives many of the hidden units' activation to be close or equal to zero, resulting in sparse connections between layers.

4.1.2. Denoising Sparse Autoencoder. In order to force the hidden layer to learn more robust features and prevent it from simply discovering the sparsity, we train a DSAE to reconstruct the input from a corrupted version of it, which is an extension of SAE. Its objective function is the same as that of SAE. The only difference is that we have to feed the corrupted input into the input layer. The structure of the DSAE is demonstrated in Figure 3. Three basic types of noise can be utilized to corrupt the input of the DSAE. The zero-masking noise [18] is employed in our model. The key idea of DSAE is to learn a sparse but robust bank of local features, which also can be called "convolution kernels." They can be used to convolve the whole image in the next convolution layer. The training procedure of the DSAE is summarized in Algorithm 1.

In this paper, we view DSAE as a "feature extractor" that takes training data X and outputs a function $f : \mathbb{R}^C \to \mathbb{R}^K$

(1) **Input**:
(2) Training set X
(3) Weight decay parameter λ, weight of sparse penalty term β, sparse parameter ρ
(4) **Procedure**:
(5) Initialize parameters (W_1, b_1), (W_2, b_2)
(6) Get $\overline{x^i}$ by stochastic corrupting the input vector x^i.
(7) **FOR** $j = 1$ to T do
(8) Loss = $L(X, Z) + \beta \sum_{j=1}^{K} \mathrm{KL}(\rho \parallel \widehat{\rho})$
(9) Use L-BFGS algorithm [58] to update (W_1, b_1), (W_2, b_2)
(10) **ENDFOR**
(11) **Output**: (W_1, b_1) which is utilized for convolution kernels

Algorithm 1: The training procedure of DSAE.

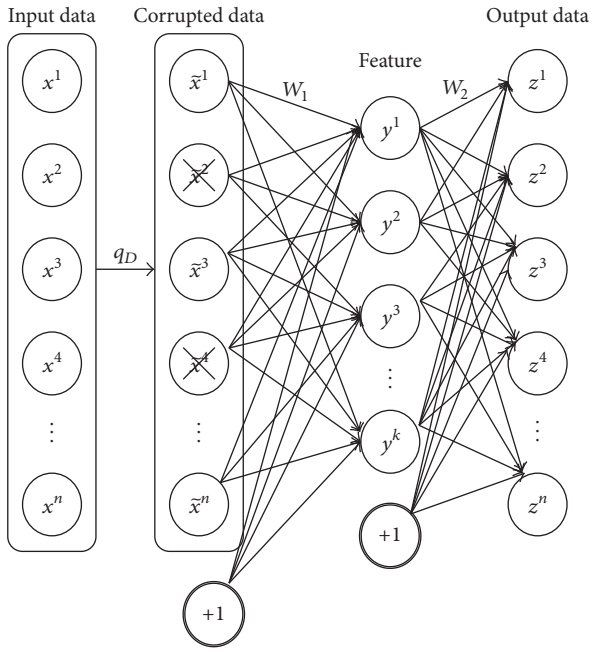


Figure 3: Illustration of a single-layer DSAE. Neurons with cross denote the corrupted input neural units.

that can map an input vector x^i to a new feature vector via the K features, where K is the number of hidden units of DSAE.

4.2. Feature Extraction. The above DSAE algorithm yields a function f that transforms an input vector $x^i \in \mathbb{R}^C$ to a new feature representation $\alpha^i = f(x^i) \in \mathbb{R}^K$. In this section, we can apply this feature extractor to our (labeled) training and testing images for classification.

4.2.1. Image Convolution. In order to extract appropriate and sufficient features from training and testing images, convolution is utilized to construct a locally connected DSAE networks. Each hidden unit connects only a small contiguous region of pixels in the input images. Sounds, natural images, and, more generally, signals that display translation invariance in any dimension can be better represented using convolutional dictionaries [53]. The convolution operator enables the system to model local structures that appear anywhere in the signal [53]. It is firstly used in natural images field by LeCun et al. [54]. Figure 4 illustrates the significance of the convolution operation. Figure 4(a) is a source image of STL-10 dataset. (b)–(d) are the convolution kernels (a.k.a. bases) trained by DSAE. (e)–(g) are the features extracted from the source image through convolution operation.

Given an image of u-by-u pixels (with F channels), we can define a $(u-w+1)$-by-$(u-w+1)$ image representation (with K channels), by using our w-by-w convolution kernel across the image with some step-size (or "stride") s equal to or greater than 1. This is illustrated in Figure 5.

4.2.2. Local Contrast Normalization. The local contrast normalization layer is inspired by computational neuroscience models [55]. It performs local subtractive and divisive normalizations, enforcing a kind of local competition between adjacent features in a feature map and between features at the same spatial location in different feature maps [56]. The subtractive normalization operation removes the weighted average of neighboring neurons from the current neuron. For a given site (i, j) of the kth feature map, it can compute

$$v_{kij} = x_{kij} - \sum_{kpg} w_{pq} \cdot x_{k,i+p,j+q}, \tag{9}$$

where w_{pq} is a Gaussian weighting window (of size 9×9 in this work) normalized so that $\sum_{kqp} w_{pq} = 1$. Based on the result of subtractive normalization, the divisive normalization computes $y_{kij} = v_{kij}/\max(\tau, \sigma_{ij})$, where $\sigma_{ij} = (\sum_{kpg} w_{pq} \cdot v_{k,i+p,j+q}^2)^{1/2}$. In our experiments, the constant τ is set to mean(σ_{ij}).

As mentioned above, we can obtain $(u - w + 1) \times (u - w + 1) \times K$ feature maps through convolution operation for a given image. Local subtractive and divisive normalizations are performed over these K feature maps by local contrast normalization layer.

4.2.3. SPP Fused with Center-Bias Prior. Bias is often highly related to the application task and sometimes can be deliberately used as a prior in a specific task to improve the

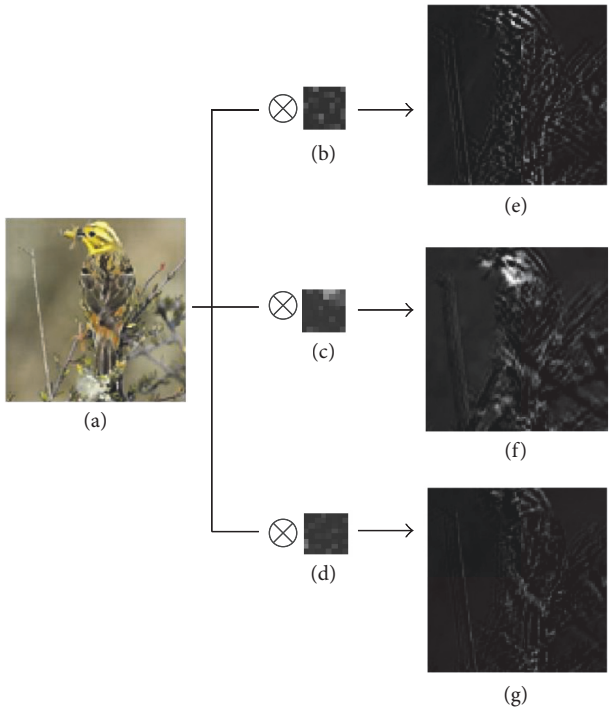


Figure 4: Examples of convolutional feature extraction. (a) is the source image. (b)–(d) are the convolution kernels learned by the single-layer DSAE. (e)–(g) are the features extracted from the source image.

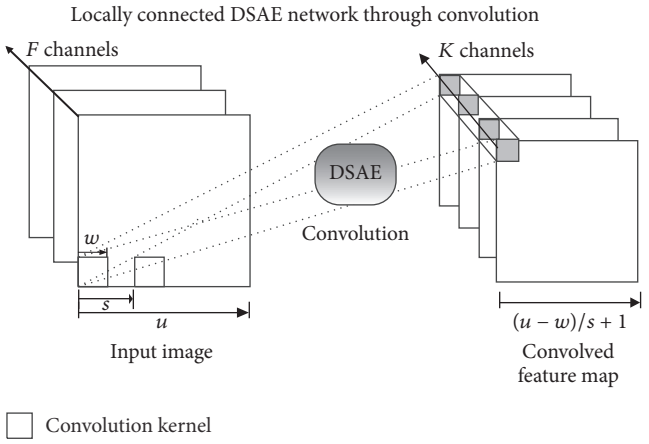


Figure 5: Illustration showing feature extraction using a w-by-w convolution kernel and a stride of s.

performance of the task [33]. When humans take pictures, they naturally tend to frame an object of interest near the center of the image. For this reason, we incorporate the center-bias prior in our work which indicates the distance to the center of each pixel. In particular, this specific prior is generated by a 2D Gaussian heatmap as showed in Figure 6.

SPP (a.k.a. spatial pyramid matching) is an extension of the Bag-of-Words (BoW) model, which is one of the most key methods in computer vision. SPP has long been an important component in the competition-winning and leading models for image classification [32]. After obtaining features using local contrast normalization as described earlier, SPP partitions the feature map into divisions from finer to coarser levels. The coarsest pyramid level has a single bin that covers the entire feature map. Figure 7(a) illustrates an example configuration of 3-level pyramid pooling (3×3, 2×2, and 1×1) about our method. In each spatial bin of every pyramid level, we pool the responses of each feature map (throughout this paper we use mean pooling). The bin sizes can be precomputed for spatial pyramid pooling. After local contrast normalization, the feature maps have a size of $a \times a$. With a pyramid level of $h \times h$ bins, we implement this pooling level as a sliding window pooling, where the window size win $= \lceil a/h \rceil$ and stride str $= \lfloor a/h \rfloor$ ($\lceil \cdot \rceil$ and $\lfloor \cdot \rfloor$ denote ceiling and floor operations). With a 3-level pyramid, we implement 3 such layers. Output of each pyramid pooling level is KM-dimensional vector with the number of bins denoted as M (K is the number of feature maps in the local contrast normalization layer). In our work, we use a 3-level pyramid (3×3, 2×2, and 1×1). So $\text{level}_{3\times3}$, $\text{level}_{2\times2}$, and $\text{level}_{1\times1}$ will generate $9K$, $4K$, and K dimensional vector, respectively.

According to the size of the three vector dimensions mentioned above, we generate the corresponding center-bias prior feature map, respectively. Then we reshape the three scales feature maps to generate column vectors, which are used for element-wise product operation with the three-dimensional column vectors after SPP. This calculation process can be showed in Figure 7 ($\odot$ denotes element-wise product between vectors in Figure 7). l_2 vector normalization is usually used to further improve FV performance [57]. The final image representation is then obtained by concatenating the results of all local column vectors from $\text{level}_{3\times3}$ to $\text{level}_{1\times1}$ (followed by l_2 vector normalization). The process of SPP fused with center-bias prior is summarized in Algorithm 2.

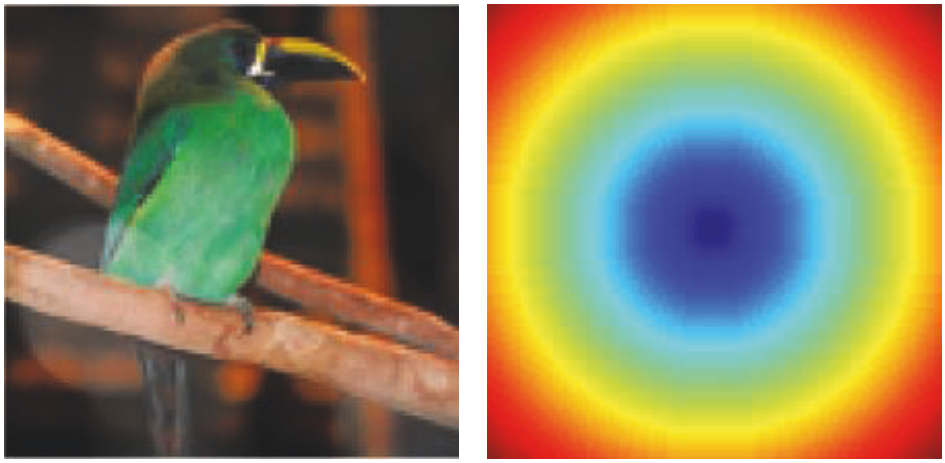
5. Overall Architecture of Image Classification

This section describes the overall architecture of the proposed method for image classification. Our method consists of four main parts, as showed in Figure 8: (1) obtaining samples for unsupervised feature learning; (2) feature learning; (3) feature extraction; and (4) classification.

(1) First, we adopt context-aware saliency detection model to compute saliency maps of image dataset, which are thresholded to get fixated and unfixated points of images. We first randomly select one image of M images and then extract a given percent of the fixations and the nonfixations. For each image, the total number of the fixations and the nonfixations extracted is N. This can be represented as a vector in $\mathbb{R}^C$ of the pixel intensity values, with $C = N \times 3$ (the inputs are natural color images). Therefore, a dataset $X \in \mathbb{R}^{C\times M}$ can thus be constructed.

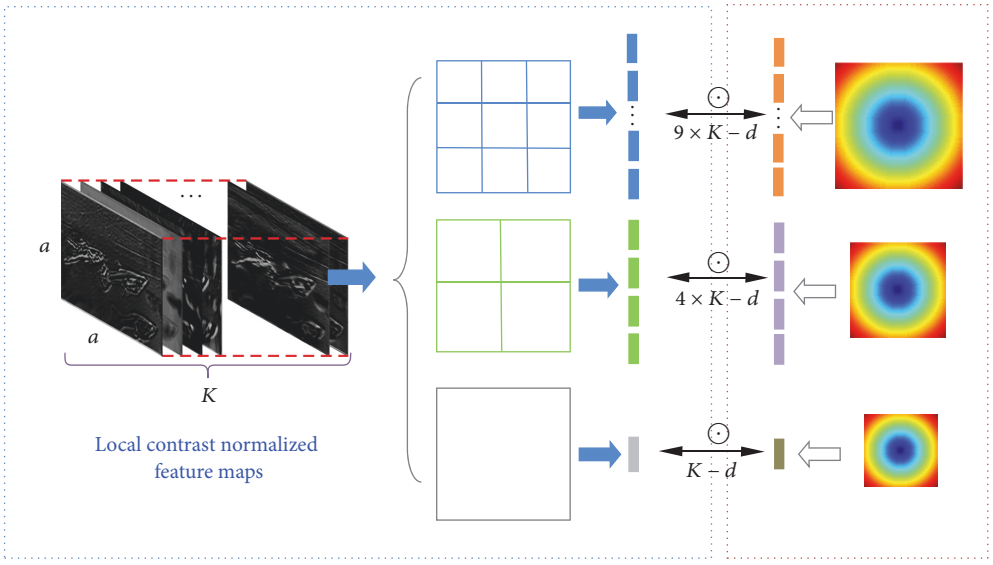
(2) Then, the dataset X is fed into a K-hidden-unit network, which is used for unsupervised feature learning of K feature extractors, according to the DSAE model.

(3) After the unsupervised feature learning, convolution is utilized to construct a locally connected DSAE networks.



(a) (b)

Figure 6: (a) A sample of STL-10 dataset. (b) Center-bias prior feature map.



(a) (b)

Figure 7: Illustration of spatial pyramid pooling fused with center-bias prior. (a) Spatial pyramid pooling layer. (b) Multiscales feature maps based on center-bias prior.

We can extract appropriate features from the training and testing images using the learned feature extractors. By using local contrast normalization method, we can increase feature sparsity and improve optimization of model. SPP fused with center-bias prior is utilized to obtain final image representation.

(4) Finally, our proposed method is combined with a linear support vector machine (SVM) to predict the label. In the case of multiclass predictions, we use the LIBLINEAR implementation for the SVM classification. It is a family of linear SVM classifiers for large-scale linear classification and an open source library which supports logistic regression and linear SVM. In our experiment, we apply L2-loss linear SVM for classification task. In addition, the regularization parameters C of the linear SVM classifier are determined by fivefold cross-validation with the arrangement of $[2^{-4}, 2^{-3}, \ldots, 2^{6}]$. A detailed description can be found in [59].

6. Experimental Setup and Results

All experiments were conducted using the computer platform of Intel® Core™ i5-4430 CPU@3.00 GHz, 32.0 GHz memory, Win 7, MATLAB R2015 (b). In order to improve the experimental operation speed, we used a parallel computing toolbox of MATLAB.

In this section, we first describe the datasets used for the experiments and display the detailed parameter settings of the proposed method. STL-10 [30] and CIFAR-10 [60] are standard datasets for unsupervised feature learning and

(1) **Input**:
(2) An input image I
(3) K feature maps after local contrast normalization layer
(4) **Procedure**:
(5) Generate center-bias prior feature maps based on I
(6) **FOR** $h \coloneqq 1$ to 3 DO
(7) For current pyramid level of $h \times h$ bins, compute win = $\lceil a/h \rceil$ and str = $\lfloor a/h \rfloor$
(8) Implement this pooling level and output $K \times h \times h$-dimensional vectors $\xi_{h\times h}$
(9) Reshape center-bias prior feature map to generate column vector $H_{h\times h}$
(10) $f_{h\times h} \longleftarrow \xi_{h\times h} \bigodot H_{h\times h}$
(11) $f_{h\times h} \longleftarrow \dfrac{f_{h\times h}}{\|f_{h\times h}\|_2}$
(12) **ENDFOR**
(13) Concatenate $f_{h\times h}$ $(1 \le h \le 3)$ to form the final spatial pyramid representation f
(14) $f \longleftarrow \dfrac{f}{\|f\|_2}$
(15) **Output** f

Algorithm 2: The pipeline of SPP fused with center-bias prior.

deep learning networks. Figure 9 shows ten examples from each image set. In this part, classification results of different models on these two datasets are showed with rigorous analysis. Then in the next part, the main techniques used in our model are evaluated with these two datasets.

6.1. Experiment and Results Analysis of STL-10 Dataset. The STL-10 dataset is a natural image set for developing deep learning and unsupervised feature learning algorithms. Each class has 500 training images and 800 testing images. The primary challenge is due to the smaller number of labeled training examples (100 per class for each training fold). Additional 10,0000 unlabeled images are provided for unsupervised learning. This dataset contains ten classes: (1) airplane; (2) car; (3) bird; (4) cat; (5) dog; (6) deer; (7) horse; (8) monkey; (9) ship; and (10) truck with a resolution of 96×96. Figure 9(a) shows some examples of STL-10 dataset. This dataset can be obtained at http://cs.stanford.edu/~acoates/stl10. We follow the standard setting in [30, 61]: (1) performing unsupervised feature learning on the unlabeled data; (2) performing supervised learning on the labeled data using predefined tenfold of 100 examples from the training data; and (3) reporting average accuracy on the full test set.

First of all, we used context-aware saliency detection method to calculate saliency maps about 100,000 unlabeled images of STL-10. Saliency maps were thresholded such that a given percent of the image pixels were classified as fixations and the rest were classified as nonfixations. For sampling of the fixated points and nonfixated points, we referred to the method of Judd et al. [35]. We chose samples from the top 5% and bottom 30% in order to have samples that were strongly positive and strongly negative; we avoided samples on the boundary between the two. We did not choose any samples within 5 pixels of the boundary of the unlabeled images. Here, we experimentally set the total number of sample points in each image equal to 64. And, in each image, the ratio of negative to positive samples was set to 1 : 4. Because the images of STL-10 dataset are RGB images, the pixel intensity value of all the collected samples in each image was expressed as the column vector $\mathbb{R}^{64*3}$. The pixel intensity value was stored in row-major order, one channel at a time. That is, the first $64 * 64$ values were the red channel, the next $64 * 64$ were green, and the last were blue. Therefore, a dataset $X \in \mathbb{R}^{192\times 100000}$ was constructed, which was subsequently fed to train DSAE.

At present, there is no perfect theoretical basis for selection of the structure of a DSAE model; we determined the optimal network structure through the experiments. To measure the classification performance with the STL-10 dataset, we first compared the classification accuracies with different number of features and sparsity parameter values. In order to study the number of features and the sensitivity of the sparsity parameter, we varied their values over a wide range. Figure 10 shows the respective performance with different number of features and sparsity parameter values. To evaluate the classification performance under different feature numbers, we considered feature representations of 400, 600, 800, 1000, and 1200 learned features. Figure 10(a) clearly shows the effect of increasing the number of learned features. The experimental analysis indicated that a feature size of 1000 produced a nice accuracy with this dataset. Based on this analysis, we set the feature size as equal to 1000 to determine the sparseness value. Figure 10(b) shows that there was a wide range of sparseness values, and the best classification performance was obtained at a sparsity value equal to 0.02. Detailed settings of other hyperparameters were set as follows: InputZeroMaskedFraction = 0.5, lambda = 0.003, beta = 5, and convolutional kernel size = $8 \times 8 \times 3$. In our experiment, we used two different 3-level pyramids: (3×3, 2×2, and 1×1) and (4×4, 2×2, and 1×1); classification result shows that the former can achieve better accuracies. In the SVM training, we intentionally did not use any data augmentation (multiview/flip). l_2 vector normalization was applied to the features for SVM training.

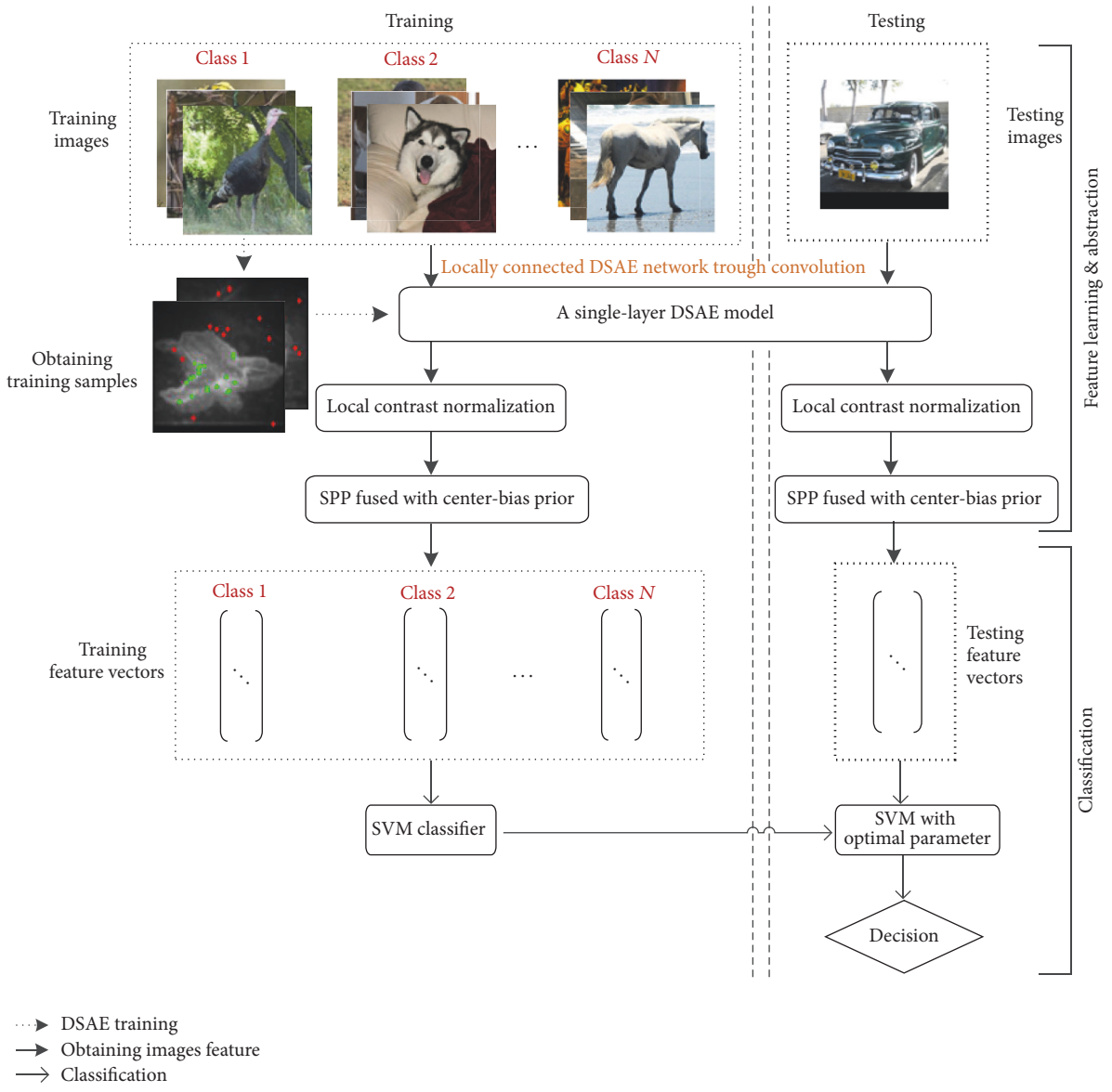
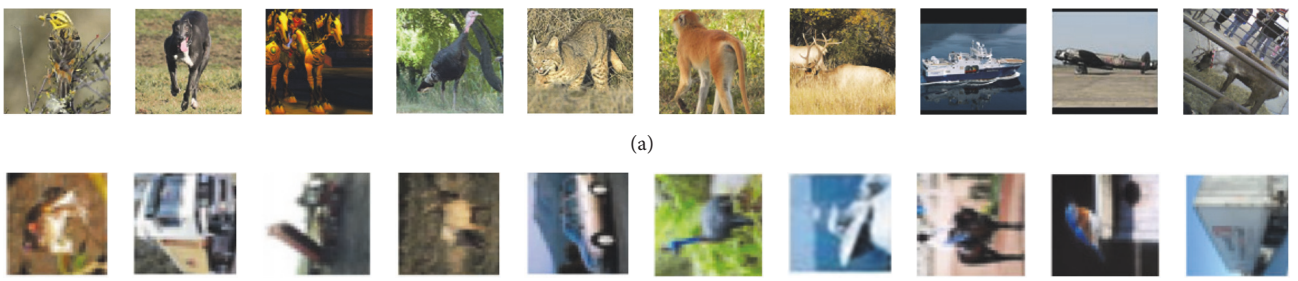


Figure 8: Overall architecture of the proposed method with all the bells and whistles.



(a)

(b)

Figure 9: Samples of the two image datasets used in our experiments. (a) STL-10. (b) CIFAR-10.

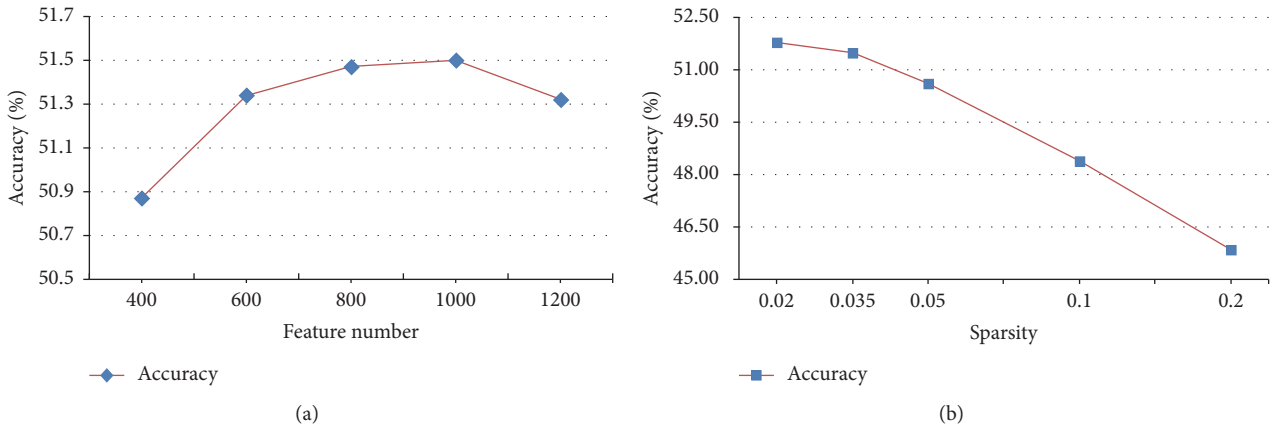


Figure 10: The effect of the feature number and sparsity parameter value on the classification accuracy with the STL-10 dataset. (a) Feature number varied over a wide range of different sizes to generate sparsity parameter. (b) Sparsity parameter value varied over a wide range.

Table 1: Comparison of average test accuracies (%) on all folds of STL-10.

Method	Accuracy
ICA (complete) [38]	48.0 ± 1.47%
Random weight baseline [38]	50.2% ± 1.08%
K-means (triangle) [30]	51.5% ± 1.73%
3 layer features from CDBN + SVM [39]	51.10%
Our method	51.8% ± 0.01%

Then, the performance of our method is compared with the previous studies on this dataset. The classification accuracy is listed in Table 1. Here, the state-of-the-art results listed for STL-10 can be improved by augmenting the training set with flip and other methods; we have not done so here and also report state of the art only for methods not doing so. Known from Table 1, we compared our single-layer model with K-means clustering algorithm—a classic single-layer network—and achieved high performance on image classification reported in [30]. Moreover, contrary to 3 layer features from CDBN + SVM [39], our shallow model shows strength in simplicity and effectiveness.

6.2. Experiment and Results Analysis of CIFAR-10 Dataset. We applied the full pipeline for CIFAR-10 which is a downsampled version of the STL-10 images (32 × 32 pixels). The CIFAR-10 dataset consists of 50,000 training images and 10,000 test images in ten classes (i.e., airplane, bird, automobile, deer, cat, frog, dog, ship, horse, and truck). These classes are completely mutually exclusive. Figure 9(b) demonstrates some examples of this dataset. Compared to STL-10 images, CIFAR-10 has a lower resolution. Hence, we achieved the total number of sample points in each image equal to 36 and convolutional kernel size was set as 6 × 6. Besides, we used all the other parameters the same as for STL-10, including inputZeroMaskedFraction, lambda, and beta. We also first compared the classification performance for varied feature numbers and sparsity parameter values in the same way as before. Figure 11 shows the classification accuracies at different feature numbers and sparsity parameter values. The results indicated that a feature size of 1200 produced the best accuracy with this dataset. Based on this analysis, for all experiments, we set the feature number equal to 1200 to generate sparsity parameter values. To evaluate the classification performance with different sparsity parameter values, we measured the overall classification accuracy for values ranging from 0.01 to 0.2. The experimental analysis showed that a sparsity parameter value of 0.02 produced an excellent accuracy with CIFAR-10. Analogous to STL-10, small values of the sparsity parameter and large feature sizes resulted in a high accuracy. This is mainly because CIFAR-10 is a downsampled version of the STL-10 images. These two dataset have similar complexity of the images.

Table 2: Comparison of accuracy (%) of the methods on CIFAR-10 with no data augmentation.

Methods	Accuracy
L2 sparse filtering [40]	63.89%
3-way factored RBM (3 layers) [41]	65.30%
Mc RBM (3 layers) [42]	71.00%
Tiled CNN [43]	73.10%
Stochastic pooling ConvNet [44]	84.87%
Deep networks with stochastic depth [45]	95.09%
Our method	74.18%

We now compare our final test results to some of the best published results on CIFAR-10. The comparison is provided in Table 2. Our method has some accuracy degradation in comparison to state-of-the-art supervised publication [45], which has increased the considerable depth of residual networks even beyond 1200 layers. The layers of 1200 are an astronomical figure. Although the performance of our fully unsupervised and extremely simple CDSAE shown here faces challenge, there is much room to exploit the dimension of network depth. Meanwhile, we still believe that our model has merits of its own. In particular, it does not require modern computers with state-of-the-art GPU or very large clusters [62] to be trained due to its simple architecture.

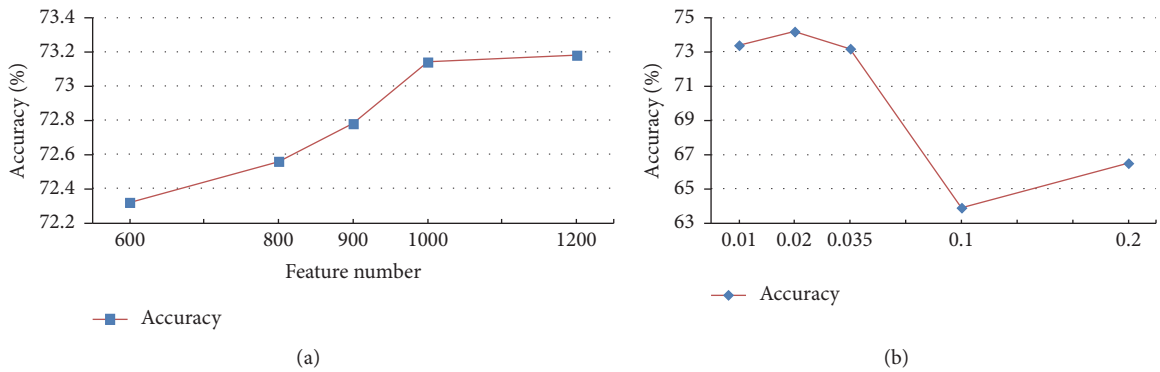


Figure 11: The effect of the feature number and sparsity parameter value on the classification accuracy with the CIFAR-10 dataset. (a) Feature number varied over a wide range of different sizes to generate sparsity parameter. (b) Sparsity parameter value varied over a wide range.

Our simple network has the advantage that information can flow efficiently forward and backward and therefore can be trained effectively and within a reasonable amount of time. Besides, it has a few hyperparameters to tune compared to increasingly complex deep models, while deeper and deeper CNN architectures have much harmful model complexity and are very difficult to train in practice. Finally, our method is a fully unsupervised feature learning method, which, though currently underperforming, still remains an appealing paradigm. It can make use of raw unlabeled images which are readily available in virtually infinite amounts. Last but not least, our model fully incorporates the theory about saliency detection and center-prior in computer vision, which are not included in the papers listed in Table 2. The performance is much larger than that on the comparable STL-10 on account of the small labeled datasets: 51.8% (±0.1%). This indicates that the model proposed here is strong when we have large labeled training sets as with CIFAR-10.

6.3. Analysis of Computational Complexity. To prove that our method is of low computational cost than some state-of-the-art methods, we focus on two representative baselines——Stochastic pooling ConvNet [44] and Deep networks with stochastic depth [45]. These two models compared are the current state-of-the-art CNNs, which outperform our method on classification accuracy. We compare the computational complexity between ours and them. The computational complexity mainly includes two parts: (1) complexity of optimizer and (2) complexity of convolutions.

Le et al. [63] introduce three off-the-shelf optimization algorithms—Stochastic Gradient Descent (SGD), Limited memory BFGS (L-BFGS), and Conjugate Gradient (CG). Our proposed methods are implemented with L-BFGS, whereas, for [44, 45], SGD is used for training. In [64], the authors demonstrate that the computational cost of SGD is $O(n)$ per iteration (where n denotes the number of variables in the system, and n below has the same definition). They also conclude that L-BFGS reduce the cost of BFGS to $O(mn)$ per iteration (where m is the number of updates allowed in L-BFGS). m is specified by the user [65]. In practice, we would rarely wish to use m greater than 15. The empirical value of m is always taken as 5, 7, and 9 [58]. Compared to a very large number of variables about n, m is much smaller. The computational cost of L-BFGS reduces to linear complexity $O(n)$.

We now turn to an analysis of complexity of convolutions. Most recently, He and Sun [66] propose the theoretical complexity of all convolutional layers. It can be represented as

$$O\left(\sum_{g=1}^{G} t_{g-1} \cdot \hbar_g^2 \cdot t_g \cdot \kappa_g^2\right), \tag{10}$$

where g is the index of a convolutional layer and G is the number of convolutional layers. t_g is the number of filters in the gth layer, and t_{g-1} is the number of input channels of the gth layer. $\hbar_g$ is the spatial length of the filter. κ_g is the spatial size of the output feature map. The fully connected layers and pooling layers often take 5–10% computational time. As a consequence, the cost of these layers is not involved in the above formulation. In our comparison, we have referred to this benchmark.

In Table 3, we have listed briefly the overall complexity of the comparison algorithms (here we consider the complexity of one iteration).

In the following, we will analyze the complexity of these models in detail.

(1) Stochastic pooling ConvNet [44] has 3 convolutional layers with 5×5 filters and 64 filter banks per layer. All of the pooling layers summarize a 3×3 neighborhood and use a stride of 2. The authors use a single fully connected layer with soft-max outputs to produce the network's class predictions. We have proved that the computational cost of SGD is $O(n)$ per iteration above. The number of variables n is evaluated as 1.3 M params (this number includes the params of convolutional layers and fully connected layer). Based on description of the model in [44], we have derived the formula $O(\sum_{g=1}^{G} t_{g-1} \cdot \hbar_g^2 \cdot t_g \cdot \kappa_g^2) = 2.1119e + 09$ ($G = 3$).

(2) Deep networks with stochastic depth [45] use the architecture described by He et al. [67] and increase the depth of residual network to 1202 layers and still yield meaningful improvements on CIFAR-10. The residual network with 1202

Table 3: Optimizer utilized and the total complexity of the models.

Model	Optimizer	Complexity	Remarks
ConvNet [44]	SGD	$O(n) + O\left(\sum_{g=1}^{G} t_{g-1} \cdot \hbar_g^2 \cdot t_g \cdot \kappa_g^2\right)$	$G = 3$; n is 1.3 M
Stochastic depth [45]	SGD	$O(n) + O\left(\sum_{g=1}^{G} t_{g-1} \cdot \hbar_g^2 \cdot t_g \cdot \kappa_g^2\right)$	$G = 1202$; n is 19.4 M
Ours	L-BFGS	$O(n) + O(t_{g-1} \cdot \hbar_g^2 \cdot t_g \cdot \kappa_g^2)$	$G = 1$; n is 0.013 M

layers has 19.4 M params [67]. However, because of lack of relative and specific parameter settings, we have to evaluate the complexity of all convolutional layers from the qualitative perspective. The authors [45] further demonstrated that very deep networks have much greater model complexity and are very difficult to train in practice and require a lot of time. Intuitively, we can infer that the complexity of this 1202-layer network is much higher than our single-layer model.

(3) In our proposed method, the dimensions of the input vector and feature are 108 and 1200, respectively. In addition, we used 6×6 filters for convolution on CIFAR-10. L-BFGS is used to train our network wherein $m \ll n$. The number of variables n here is calculated as 0.013 M parameters. From this it could be suggested that, with comparison of $O(n)$, our complexity of optimizer is lower than [44, 45]. For convolutional complexity of our model, we have calculated the corresponding computational cost as follows: $O(\sum_{g=1}^{G} t_{g-1} \cdot \hbar_g^2 \cdot t_g \cdot \kappa_g^2) = O(t_{g-1} \cdot \hbar_g^2 \cdot t_g \cdot \kappa_g^2) = 9.4478e + 07$ ($G = 1$).

Based on the details analyzed above, it is indicated that our method has low computational cost than the compared state-of-the-art methods.

6.4. Analysis of CDAE's Properties. In this section, we mainly analyze the influence of techniques and structures designed in the proposed algorithm. The key structures that contribute to the success of our network are local contrast normalization layer, SPP fused with center-bias prior, and l_2 vector normalization. We evaluate the impact of each of these three improvements considered separately.

Impact of Local Contrast Normalization Layer. We start by studying the influence of the local contrast normalization layer, which is a single but important ingredient for good accuracy on object recognition benchmarks [56]. We note that local contrast normalization is key to obtaining good results: without it, the accuracy is 50.78% (±0.6%) for STL-10 and 72.22% for CIFAR-10. While adding it, the accuracy can be improved around 1% and 2%, respectively.

Impact of Center-Bias Prior. People use a lot of prior knowledge in interpretation of an image; prior knowledge can be used for a specific task to improve its performance [68]. SPP fused with center-bias prior is efficient in image classification: it raises the accuracy of STL-10 from 49.18% to 51.8% and CIFAR-10 from 74.01% to 74.18%. On the other hand, the SPP fused with center-bias prior slightly raises the benchmark for CIFAR-10: an intuitive interpretation is that images of CIFAR-10 have lower resolution compared to STL-10. It is advantageous in datasets with high resolution.

Impact of l_2 Vector Normalization. We now evaluate the influence of the l_2 vector normalization of high-dimensional vectors before using SVM training. l_2 vector normalization improves performance in these two datasets by 5% on STL-10 (46.92% $\rightarrow$ 51.8%) and CIFAR-10 (68.31% $\rightarrow$ 74.18%). We see that the l_2 vector normalization is powerful, which can improve classification results over no normalization dramatically. Through experiments, we show that these three complementary factors elevate the classification accuracy of our CDSAE. They are all indispensable to our model as there is usually a big drop in accuracy when removing these structures.

7. Conclusion and Future Work

In this paper, an arguably simple but effective image classification approach called CDSAE is proposed. It is an improvement of the existing successful networks DAE, SAE, and CNN. CDSAE efficiently integrates the following components: combining DAE and SAE to construct DSAE, embedding local contrast normalization layer following convolution operation, and, most importantly, building a spatial pyramid pooling fused with center-bias prior in a natural way. CDSAE has superiority in low computational cost and fewer amounts of hyperparameters to tune, while only suffering from reduced performance relative to some state-of-the-art methods.

In experiment, we find that the following are particularly imperative.

(1) Local Contrast Normalization. It shows greater effectiveness in improving performance compared to not using it.

(2) Center-Bias Prior. It can effectively capture the center-prior information of datasets, which is particularly appropriate for object-centered images with high resolution.

(3) l_2 Vector Normalization. l_2 vector normalization is more effective than nonnormalization.

In our future research, more investigations can be done on the proposed framework. Firstly, we plan to extend this approach to learn hierarchical features of images from low-level to high-level feature representation. Secondly, center-bias prior is more specifically suited to the classified object

which is in the center of an image. Other more general prior knowledge about images can be further introduced into the framework.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work is partially supported by the National Natural Science Foundation of China (Grant nos. 61603326, 61602400) and Natural Science Foundation of Yancheng Teachers University (Grant nos. 15YCKLQ011 and 15YCKLQ012).

References

[1] L. Shao, L. Liu, and X. Li, "Feature learning for image classification via multiobjective genetic programming," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 7, pp. 1359–1371, 2014.

[2] H. Yin, X. Jiao, Y. Chai, and B. Fang, "Scene classification based on single-layer SAE and SVM," *Expert Systems with Applications*, vol. 42, no. 7, pp. 3368–3380, 2015.

[3] Y. Luo, Y. Wen, D. Tao, J. Gui, and C. Xu, "Large margin multimodal multi-task feature extraction for image classification," *IEEE Transactions on Image Processing*, vol. 25, no. 1, pp. 414–427, 2016.

[4] J. F. Serrano-Talamantes, C. Avilés-Cruz, J. Villegas-Cortez, and J. H. Sossa-Azuela, "Self organizing natural scene image retrieval," *Expert Systems with Applications*, vol. 40, no. 7, pp. 2398–2409, 2013.

[5] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[6] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05)*, vol. 1, pp. 886–893, June 2005.

[7] T. Ahonen, A. Hadid, and M. Pietikäinen, "Face recognition with local binary patterns," in *Proceedings of the European Conference on Computer Vision (ECCV)*, vol. 3021, pp. 469–481, Prague, Czech Republic, 2004.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the 26th Annual Conference on Neural Information Processing Systems (NIPS '12)*, pp. 1097–1105, Lake Tahoe, Nev, USA, December 2012.

[9] J. Schmidhuber, "Deep learning in neural networks: an overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[10] J. Yang, K. Yu, Y. Gong, and T. Huang, "Linear spatial pyramid matching using sparse coding for image classification," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '09)*, pp. 1794–1801, June 2009.

[11] G. E. Hinton, "Deep belief networks," *Scholarpedia*, vol. 4, no. 5, article 5947, 2009.

[12] Y. Bengio, "Learning deep architectures for AI," *Foundations and Trends in Machine Learning*, vol. 2, no. 1, pp. 1–27, 2009.

[13] K. Fukushima, "Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, no. 4, pp. 193–202, 1980.

[14] J. Weng, N. Ahuja, and T. Huang, "Cresceptron: a self-organizing neural network which grows adaptively," in *Proceedings of the IJCNN International Joint Conference on Neural Networks*, pp. 576–581, Baltimore, MD, USA, 1992.

[15] H. Bourlard and Y. Kamp, "Auto-association by multilayer perceptrons and singular value decomposition," *Biological Cybernetics*, vol. 59, no. 4-5, pp. 291–294, 1988.

[16] H.-C. Shin, M. R. Orton, D. J. Collins, S. J. Doran, and M. O. Leach, "Stacked autoencoders for unsupervised feature learning and multiple organ detection in a pilot study using 4D patient data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1930–1943, 2013.

[17] A. Ng, *Sparse Autoencoder*, vol. 72 of *CS294A Lecture Notes*, 2011.

[18] P. Vincent, H. Larochelle, Y. Bengio, and P. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th International Conference on Machine Learning*, pp. 1096–1103, ACM, Helsinki, Finland, July 2008.

[19] J. Xu, L. Xiang, R. Hang, and J. Wu, "Stacked Sparse Autoencoder (SSAE) based framework for nuclei patch classification on breast cancer histopathology," in *Proceedings of the 2014 IEEE 11th International Symposium on Biomedical Imaging, ISBI 2014*, pp. 999–1002, Beijing, China, May 2014.

[20] P. Vincent, H. Larochelle, I. Lajoie, and P. Manzagol, "Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion," *Journal of Machine Learning Research*, vol. 11, pp. 3371–3408, 2010.

[21] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber, "Stacked convolutional auto-encoders for hierarchical feature extraction," in *Proceedings of the International Coriference on Artificial Neural Networks (ICANN)*, pp. 52–59, Springer, Berlin, Germany, 2011.

[22] E. Nowak, F. Jurie, and B. Triggs, "Sampling strategies for bag-of-features image classification," in *Computer Vision—ECCV 2006*, A. Leonardis, H. Bischof, and A. Pinz, Eds., vol. 3954 of *Lecture Notes in Computer Science*, pp. 490–503, Springer, Berlin, Germany, 2006.

[23] A. A. Salah, E. Alpaydin, and L. Akarun, "A selective attention-based method for visual pattern recognition with application to handwritten digit recognition and face recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 3, pp. 420–425, 2002.

[24] A. Borji and L. Itti, "State-of-the-art in visual attention modeling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 185–207, 2013.

[25] E. Mazurova, *Accuracy of Measurements of Eye-Tracking of a human perception on the screen*, Degree thesis, Department of International Business, Arcada - Nylands svenska yrkeshögskola (2014).

[26] E. Erdem and A. Erdem, "Visual saliency estimation by nonlinearly integrating features using region covariances," *Journal of Vision*, vol. 13, no. 4, article 11, 2013.

[27] P.-H. Tseng, R. Carmi, I. G. M. Cameron, D. P. Munoz, and L. Itti, "Quantifying center bias of observers in free viewing of dynamic natural scenes," *Journal of Vision*, vol. 9, no. 7, article 4, 2009.

[28] L. Zhang, M. H. Tong, T. K. Marks, H. Shan, and G. W. Cottrell, "SUN: a Bayesian framework for saliency using natural statistics," *Journal of Vision*, vol. 8, no. 7, article 32, 2008.

[29] B. W. Tatler, "The central fixation bias in scene viewing: Selecting an optimal viewing position independently of motor biases and image feature distributions," *Journal of Vision*, vol. 7, no. 14, article 4, 2007.

[30] A. Coates, A. Y. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning in," in *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 215–223, 2011.

[31] F. Zhang, B. Du, and L. Zhang, "Saliency-guided unsupervised feature learning for scene classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 4, pp. 2175–2184, 2015.

[32] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1904–1916, 2015.

[33] A. Borji, M. M. Cheng, H. Jiang, and J. Li, "Salient object detection: A survey," https://arxiv.org/abs/1411.5878.

[34] S. Wang, J. Zhang, and C. Zong, "Learning Sentence Representation with Guidance of Human Attention," https://arxiv.org/abs/1609.09189.

[35] T. Judd, K. Ehinger, F. Durand, and A. Torralba, "Learning to predict where humans look," in *Proceedings of the 12th International Conference on Computer Vision (ICCV '09)*, pp. 2106–2113, IEEE, Kyoto, Japan, October 2009.

[36] W. Kienzle, F. A. Wichmann, B. Schölkopf, and M. O. Franz, "A nonparametric approach to bottom-up visual saliency," in *Proceedings of the 20th Annual Conference on Neural Information Processing Systems, NIPS 2006*, pp. 689–696, MIT Press, Vancouver, Canada, December 2006.

[37] S. Goferman, Z. M. Lihi, and A. Tal, "Context-aware saliency detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 10, pp. 1915–1926, 2012.

[38] J. Ngiam, Z. Chen, and A. S. Bhaskar, "Sparse filtering," *Advances in Neural Information Processing Systems*, pp. 1125–1133, 2011.

[39] J. Lee, J. H. Lim, H. Choi, and D. S. Kim, "Multiple Kernel Learning with Hierarchical Feature Representations," in *Proceedings of the International Conference on Neural Information Processing (ICNIP)*, pp. 517–524, Springer, Berlin, Germany, 2013.

[40] Z. Yang, L. Jin, D. Tao, S. Zhang, and X. Zhang, "Single-layer Unsupervised Feature Learning with L2 regularized sparse filtering," in *Proceedings of the 2nd IEEE China Summit and International Conference on Signal and Information Processing, IEEE ChinaSIP 2014*, pp. 475–479, July 2014.

[41] A. Krizhevsky and G. E. Hinton, "Factored 3-way restricted boltzmann machines for modeling natural images," in *Proceedings of the International conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 621–628, 2010.

[42] M. Ranzato and G. E. Hinton, "Modeling pixel means and covariances using factorized third-order Boltzmann machines," in *Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2010*, pp. 2551–2558, San Francisco, Calif, USA, June 2010.

[43] J. Ngiam, Z. Chen, D. Chia, P. W. Koh, Q. V. Le, and A. Y. Ng, "Tiled convolutional neural networks," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, pp. 1279–1287, Vancouver, Canada, 2010.

[44] M. D. Zeiler and R. Fergus, "Stochastic pooling for regularization of deep convolutional neural networks," in *Proceedings of the 1nd International Conference on Learning Representations (ICLR)*, Scottsdale, Ariz, USA, 2013.

[45] G. Huang, Y. Sun, and Z. Liu, "Deep networks with stochastic depth," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 646–661, Springer International Publishing, Amsterdam, Netherlands, 2016.

[46] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2323, 1998.

[47] Y. Boureau, J. Ponce, and Y. Lecun, "A theoretical analysis of feature pooling in visual recognition," in *Proceedings of the 27th International Conference on Machine Learning (ICML '10)*, pp. 111–118, Haifa, Israel, June 2010.

[48] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proceedings of the 30th International Conference on Machine Learning (ICML)*, Atlanta, GA, USA, 2013.

[49] H. H. Aghdam, E. J. Heravi, and D. Puig, "Recognizing Traffic Signs Using a Practical Deep Neural Network," in *Proceedings of the Robot 2015: Second Iberian Robotics Conference (ROBOT)*, pp. 399–410, Springer, Lisbon, Portugal, 2016.

[50] C. Zhang and P. C. Woodland, "Parameterised sigmoid and ReLU hidden activation functions for DNN acoustic modelling," in *Proceedings of the 16th Annual Conference of the International Speech Communication Association, INTERSPEECH 2015*, pp. 3224–3228, Dresden, Germany, September 2015.

[51] H. Lee, C. Ekanadham, and A. Y. Ng, "Sparse deep belief net model for visual area V2," in *Proceedings of the Advances in neural information processing systems (NIPS)*, pp. 873–880, MIT Press, 2008.

[52] S. Kullback and R. A. Leibler, "On information and sufficiency," *Annals of Mathematical Statistics*, vol. 22, pp. 79–86, 1951.

[53] K. Kavukcuoglu, P. Sermanet, Y.-L. Boureau, K. Gregor, M. Mathieu, and Y. L. LeCun, "Learning convolutional feature hierarchies for visual recognition," in *Proceedings of the 24th Annual Conference on Neural Information Processing Systems (NIPS '10)*, pp. 1090–1098, Curran Associates, Inc., Vancouver, Canada, December 2010.

[54] Y. LeCun, B. Boser, J. S. Denker et al., "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[55] N. Pinto, D. D. Cox, and J. J. DiCarlo, "Why is real-world visual object recognition hard?" *PLoS Computational Biology*, vol. 4, no. 1, pp. 0151–0156, 2008.

[56] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, "What is the best multi-stage architecture for object recognition?" in *Proceedings of IEEE 12th International Conference on Computer Vision (ICCV '09)*, pp. 2146–2153, Kyoto, Japan, October 2009.

[57] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep fisher networks for large-scale image classification," in *Proceedings of the Advances in neural information processing systems (NIPS)*, pp. 163–171, Curran Associates, South Lake Tahoe, Calif, USA, 2013.

[58] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Mathematical Programming*, vol. 45, no. 3, pp. 503–528, 1989.

[59] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: A library for large linear classification," *Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008.

[60] A. Krizhevsky, *Learning Multiple Layers of Features from Tiny Images [M.S., thesis]*, Department of Computer Science, University of Toronto, 2009.

[61] A. Coates and A. Y. Ng, "Selecting receptive fields in deep networks," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, pp. 2528–2536, Granada, Spain, 2011.

[62] Q. V. Le, "Building high-level features using large scale unsupervised learning," in *Proceedings of the 38th IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '13)*, pp. 8595–8598, Vancouver, Canada, May 2013.

[63] Q. V. Le, J. Ngiam, A. Coates, A. Lahiri, B. Prochnow, and A. Y. Ng, "On optimization methods for deep learning," in *Proceedings of the 28th International Conference on Machine Learning (ICML '11)*, pp. 265–272, Bellevue, Wash, USA, July 2011.

[64] N. N. Schraudolph, J. Yu, and S. Günter, "A stochastic quasi-Newton method for online convex optimization," in *Proceedings of the International Conference on Intelligence and Statistics (AISTATS)*, pp. 436–443, San Juan, Puerto Rico, 2007.

[65] T. N. Sainath, L. Horesh, B. Kingsbury, A. Y. Aravkin, and B. Ramabhadran, "Accelerating Hessian-free optimization for Deep Neural Networks by implicit preconditioning and sampling," in *Proceedings of the 2013 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2013*, pp. 303–308, Olomouc, Czech Republic, December 2013.

[66] K. He and J. Sun, "Convolutional neural networks at constrained time cost," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '15)*, pp. 5353–5360, Boston, Mass, USA, June 2015.

[67] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*, pp. 770–778, Las Vegas, Nev, USA, July 2016.

[68] H. Z. Ai and Y. C. Su, *Image Processing, Analysis, and Machine Vision*, Tsinghua University Press, Beijing, China, 2011.