Learning Deep Hierarchical Visual Feature Coding

Hanlin Goh, Nicolas Thome, *Member, IEEE,* Matthieu Cord, *Member, IEEE,* and Joo-Hwee Lim, *Member, IEEE*

***Abstract*—In this paper, we propose a hybrid architecture that combines the image modeling strengths of the Bag of Words framework with the representational power and adaptability of learning deep architectures. Local gradient-based descriptors, such as SIFT, are encoded via a hierarchical coding scheme composed of spatial aggregating restricted Boltzmann machines (RBM). For each coding layer, we regularize the RBM by encouraging representations to fit both sparse and selective distributions. Supervised fine-tuning is used to enhance the quality of the visual representation for the categorization task. We performed a thorough experimental evaluation using three image categorization datasets. The hierarchical coding scheme achieved competitive categorization accuracies of 79.7% and 86.4% on the Caltech-101 and 15-Scenes datasets, respectively. The visual representations learned are compact and the model's inference is fast, as compared to sparse coding methods. The low-level representations of descriptors that were learned using this method result in generic features that we empirically found to be transferrable between different image datasets. Further analysis reveal the significance of supervised fine-tuning when the architecture has two layer of representations as opposed to a single layer.**

***Index Terms*—Computer Vision, Image Categorization, Hierarchical Visual Architecture, Bag-of-Words (BoW) Framework, Sparse Feature Coding, Dictionary Learning, Restricted Boltzmann Machine (RBM), Deep Learning, Transfer Learning**

I. Introduction

ONE key challenge in computer vision is the problem of image categorization, which involves predicting semantic categories, such as scenes or objects, from the raw pixels of images. While the solution to bridge this semantic gap remains elusive, promising developments have been proposed that stride towards this goal.

In the last decade, Bag of Words (BoW) frameworks [1] have achieved good classification performances on many object and scene image data sets. It consists of four major steps (Fig. 1), namely: 1) descriptor extraction, 2) feature coding, 3) spatial pooling and 4) SVM classification, to classify an image into its semantic category. In a typical setup, gradient-based local image descriptors, such as scale-invariant feature transform (SIFT) [2] and histogram of orientated gradients (HOG) [3], are used to describe an image. They are discriminative yet robust to various image degradations. A common adaptation for image categorization is the use of spatial pyramids [4] to integrate spatial information. However, in the classical formulation, the feature coding step is generally a fixed and flat single layer operation. The flat structure limits the representational power of model, while the lack of learning makes it difficult to adapt to different data.

Recently, in a separate research direction, the convolutional deep neural network [5] has emerged as a competitive method for classifying large-scale image datasets with huge amounts of training data [6]. Due to the depth and plasticity of these networks, the variety of information it can learn to represent is extensive. This network is fully-supervised and requires a lot of labeled training data to perform well and avoid over fitting. It is not clear that it is able to learn a meaningful representations for categorizing moderate-sized datasets, with fewer labeled training examples.

In this paper, we propose a hybrid hierarchical architecture based on restricted Boltzmann machines (RBM) to encode SIFT descriptors and provide the vectorial representation for image categorization. The hybrid architecture merges the complementary strengths of the BoW framework and deep architectures. In particular, we exploit the modeling power of local descriptors and spatial pooling of the BoW framework, and the adaptability and representational power of deep learning. Table I details the features of our method against other relevant methods. Our main technical contributions are as follows:

- We extend our previous work on coding SIFT descriptors [7] from a flat operation to a deep architecture. This exploit the representational power of network depth to gradually bridge the semantic gap for image categorization. To our knowledge, ours is the first deep learning model using RBMs to encode SIFT descriptors. We also exploit spatial information by aggregating representations within our model.
- In contrast to other dictionary learning methods, our dictionaries are regularized to be jointly sparse and selective based on power-law distributions. Following our preliminary studies [7], both RBM layers are penalized with respect to individual visual codewords and their corresponding input example. The entire hierarchical visual dictionary is subsequently fine-tuned with deep supervised signals.
- On both the Caltech-101 and 15-Scene datasets, our hierarchical architecture achieves competitive categorization performances among the family of feature coding methods embedded in a standard BoW framework. Inference is also faster than sparse coding methods. Our comprehensive experimental analyses show the importance of supervised fine-tuning for hierarchical architectures. We also analyzed the possibility of transferring shallow and deep representations across different datasets.

Manuscript received April 11, 2013; accepted February 14, 2014.

H. Goh and J.-H. Lim are with the Institute for Infocomm Research, A*STAR, Singapore and the Image and Pervasive Access Laboratory, CNRS UMI 2955, France-Singapore. E-mail: hlgoh@i2r.a-star.edu.sg & joohwee@i2r.a-star.edu.sg

N. Thome and M. Cord are with the Laboratoire d'Informatique de Paris 6, UPMC – Sorbonne Universitès, Paris, France. E-mail: nicolas.thome@lip6.fr & matthieu.cord@lip6.fr

This work was supported in part by the French Embassy in Singapore through the Merlion Project and Ph.D. Program from 2010 to 2012.

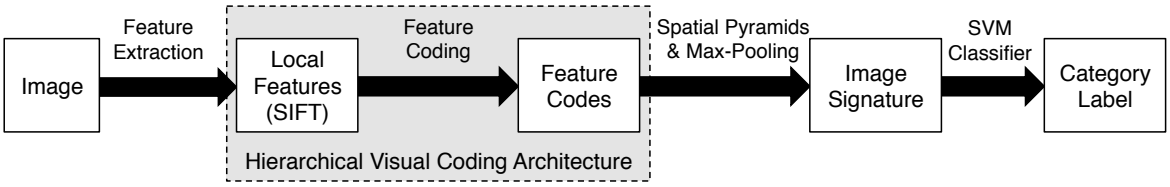


Fig. 1. A sequence of processes in the Bag of Words framework takes us from the image representation to the category label. The hierarchical visual coding architecture (gray) is responsible for transforming local features into feature codes.

TABLE I
FEATURE COMPARISON WITH RELATED METHODS

Properties	CRBM [8]	CDBN [9]	Sparse Coding [10]	Our Architecture
SIFT input	✓		✓	✓
Deep		✓		✓
Sparse			✓	✓
Selective	✓	✓		✓
Supervised				✓

The rest of this paper is organized as follows. Section II introduces the evolution of methods for learning deep neural networks, as well as related work in the BoW framework with a focus on the feature coding step. We detail our methodology to construct the hierarchical architecture to model image descriptors in Section III and our learning algorithm in Section IV. In Section V, we present our experimental results on image categorization together with comprehensive analyses using various experimental datasets. Finally, Section VI concludes the paper with suggestions of future work.

II. Related Work

A. Learning Deep Architectures for Vision

A hierarchical architecture consists of multiple layers combined as a series of basic operations. The architecture takes raw input data at the lowest level and processes them via a sequence of basic computational units until the data is transformed to a suitable representation in the higher layers. Multiple layers of distributed coding allows the network to encode highly varying functions efficiently [11]. When there are three or more layers exist in the architecture, it is considered to be deep [12], [13]. The learning of deep architectures has emerged, as an effective framework for modeling complex relationships among high-dimensional structured data. It learns a hierarchy of meaningful representations that carry some intrinsic value for classification tasks. As a result, deep learning methods have been applied to a variety of tasks, in domains such as vision, audio and language processing.

The current methods for learning deep architectures for vision is a cumulation of much research over the years. Neural networks, popularized in the 1980s, revolutionized the notion of learning from data. In particular, fully-connected multi-layered perceptrons, having been shown to be universal approximators, can represent any function with its parameters [14]. However, researchers often have to deal with the problems of a huge number of parameters and the difficult non-convex optimization problem. The optimization is even more tedious for networks that are deep.

To tackle the vision problem, the convolutional neural network [15], [16] was developed. It is a fully-supervised multi-layered network with convolution operators in each layer mapping their inputs to produce a new representation via a bank of filters. Such a highly adapted hierarchical local connectivity has the potential to encode structure suitable for modeling images, such that even with random weights in the early layer, performance remains impressive [17]. Unsurprisingly, it has produced exceptional performances for specific image datasets [15], [16]. However, being a fully-supervised model, learning often gets stuck in local minima and special care must be given to handle the network depth.

Neural networks are black boxes that are difficult to understand and train. As such, there was a lost in interest in the neural networks in the 1990s, as more started to favor kernel methods, such as the support vector machine (SVM). SVMs treat the learning problem as a convex optimization and are easy to train. However, for SVMs to work well, users need to provide complex handcrafted features or design suitable kernels for classification. They also have limited representational power due to its flat architecture.

The problem of training deep neural networks was recently given a new lease of life with a focus on unsupervised learning. This emphasis is crucial because there is usually a lot more unlabeled data compared to labeled ones. The solution considers each layer as an unsupervised generative gradient-based module that learns from its input distribution and stacks them one layer at a time from the bottom-up, in a greedy manner [12], [13], [18]. This makes it scale well to large networks. It also appears sensible to learn simple representations first and higher-level abstractions on top of existing lower-level ones. In place of randomly initialized parameters, this unsupervised representation forms the initialization – a catalyst to learn meaningful representations – for the subsequent supervised discriminative learning phase.

There are three popular methods to learning a network of fully-connected layers, namely: deep belief networks (DBN), deep autoencoder, deep sparse coding. The DBN [12], which greedily stacks layers of restricted Boltzmann machines (RBM) [19], each trained to minimize an energy function that maximizes the likelihood of its input distribution. Details of RBM training is explained in Section IV. Supervision can be introduced to train the network by running the error backpropagation algorithm [20], [21] across all layers in a separate training phase [22]. A similar but simpler architecture

known as the deep autoencoder, replaces the RBM building block of the DBN with a two-layer autoencoder network, which tries to get back the original inputs at its output layer, hence minimize the input reconstruction error [13], [23]. The variants of this architecture focus on avoiding trivial solutions, such as the identity. The input reconstruction error is also used, though in a slightly different manner, in sparse dictionary learning [24], which can also be hierarchically stacked to become a deep network. Sparse dictionary learning has also gained popularity for encoding image features in vision [25]. To perform sparse coding, we consider a set of input vectors $\mathbf{x} \in \mathbb{R}^I$ and a projection matrix $\mathbf{W} \in \mathbb{R}^{I \times J}$ containing the set of J codewords. The optimization attempts to find the vector of linear projections $\mathbf{z} \in \mathbb{R}^J$ that explicitly minimizes the feature reconstruction error, along with a regularization term that promotes sparse solutions:

$$\mathbf{z}^* = \underset{\mathbf{z}}{\operatorname{argmin}} \|\mathbf{x} - \mathbf{W}\mathbf{z}\|_2^2 + \lambda \|\mathbf{z}\|_1 \,, \tag{1}$$

where λ is a constant and the ℓ_1 norm is often used to approximate the ℓ_0 norm, leading to a convex problem for $\mathbf{W}$ given a fixed $\mathbf{z}$, or vice versa.

With the emergence of these foundational deep learning methods, the computer vision community now have a new set of tools to apply. Conventional deep learning methods are fully-connected models, which poses a problem when scaling up to large images images. Lee et al. [9] exploited the learning capabilities of the RBM-based DBN and extended it by adding convolutional and pooling operators to learn to classify large images from the pixel-level. However, results fell short of the state-of-the-art performances by variants of the BoW framework for image classification.

Most recently, convolutional deep neural networks [5] have recently emerged as the best performing model in the Large Scale Visual Recognition Challenge using the ImageNet database [6], a large-scale dataset with 1,000,000 labeled training images. The network consists of seven learned layers (five convolutional and two fully-connected) that map image pixels to the semantic-level. The model is trained in an entirely supervised manner using stochastic gradient descent with the backpropagation algorithm. Despite this good performances, it has to rely on many training examples to perform well and has not been shown to work as well for moderate-sized dataset, with relatively fewer labeled data. Another deep architecture that exploits the KSVD method for learning has also been shown to perform well for object recognition [26].

In general, the deep learning methods presented perform extremely well on modeling input data. Here, we exploit the learning ability and representational power of deep learning to form a hierarchy of latent representations of gradient-based SIFT descriptors [2], extracted from local patches in the image.

B. Bag of Words Framework for Image Categorization

The BoW pipeline of four successive modules (Fig. 1) provides state-of-art results on many image categorization problems. It describe the image with a robust set of gradient-based local descriptors, SIFT [2] and HOG [3]. These descriptors are invariant to various image degradations, such as geometric and photometric transformations, which is essential when addressing image categorization problems.

A crucial aspect of the BoW framework is the transformation from the set of local descriptors to a constant-sized image vector used for classification. Converting the set of local descriptors into the final vectorial image representation is performed by a succession of two steps: coding and pooling. The coding step transforms the input features into a representation of visual words. Due to visual word ambiguity [27], this coding step has garnered much attention by those trying to capture meaningful representations.

A main shortcoming is that coding is generally a fixed single-layer operation. The flat structure limits the representational power of the model, while the lack of learning involved makes kit unable to adapt well to the vision tasks. There is extensive work studying the coding of local descriptors and a growing interest in applying machine learning techniques to improve this process. To tackle the dictionary learning problem, some groups focus on unsupervised techniques, such as sparse dictionary learning [10], [25], [28] and restricted Boltzmann machines (RBM) [8], [29], while others rely on supervised learning [30], [31], [32], [33], [34].

The sparse dictionary learning method makes it possible to learn the mapping of input descriptors to sparse representations. However, it is a decoder network, where the sparse coding optimization (Eq. 1) needs to be solved for every descriptor. This makes inference very slow, especially when there are many descriptors or when the dictionary is large. There are various approximations that help reduce the computational costs, such as limiting the optimization to within a component of a mixture model [35], using a small dictionary for sparse coding and another for pooling [28] or incorporating locality constraints [36]. A more relevant approach is to learn an encoding-decoder networks [25], in which an encoder is concurrently learned to avoid performing the heavy minimization step of Eq. 1 during inference.

Another type of encoder-decoder network is the RBM, which is faster than traditional sparse coding during inference. Sohn et al. [8] used Gaussian RBMs to learn representations from SIFT, but the overall architecture is heavy and difficult to train. Our previous work focused on manipulating RBM representations to get desirable representations [37], such as topographic maps for transformation invariance [38]. We extended this to a preliminary study that encoded SIFT using a shallow RBM dictionary within the BoW framework [7]. In Sections IV-A and IV-B, we point out some drawbacks of existing regularization methods [9] and suggest a method to regularize RBMs with selectivity and sparsity.

Since unsupervised deep learning forms the initialization step for supervised learning, it is important to review supervised feature coding methods that increase class separation through discriminative loss functions, such as mutual information loss [39]. The methods may be classified with respect to the scale in which image labels are incorporated. Some methods directly use a discriminative criterion for each local descriptor [30], [31]. However, an image label usually does not propagate to every local portion of the image, so these methods do not pose the exact image classification problem and are

more likely to suffer from noise. Other methods associate labels to a spatially-pooled global image statistic [32], [33], [34]. This information has to be "un-pooled" for dictionary optimization, making the methods complex and slow. With deep architectures, it is necessary to implement a supervised scheme to optimize classification performances (Section IV-C).

In this work, feature coding is done using a hierarchy of RBMs, with dictionaries learned via a combination of unsupervised and supervised learning. Our experiments show that this leads to better higher-level visual representations and improved image categorization performances.

After coding, the pooling step constructs a vectorial representation from the set of local features. To attenuate the loss of spatial information when aggregating over the image, the spatial pyramid scheme [4] pools representations from image regions defined by a multi-resolution spatial grid. Recent work show that max-pooling outperforms the traditional average-pooling, especially when linear classifiers are used [34], [40]. While other work studying the pooling beyond a scalar value [41], [42], [43], [44] have shown promising results. However, their representation dimensionality, typical in the millions, is huge. Some recent work also focus combining coding and pooling, blurring the lines between the two operations [45]. In contrast, the representation of our architecture is compact, even compared to other BoW methods. Finally, BoW-based methods generally rely on linear support vector machines (SVM) [46] to solve the classification problem.

III. Constructing Hierarchical Visual Architectures

The four crucial steps of the BoW framework (Fig. 1) are namely, the extraction of local descriptors, local feature coding, spatial pooling to form the image signature and image classification using SVMs. In the process, the representation is transformed from a low-dimensional but high-cardinality representation to a single high-dimensional vector that describes the entire image. This is illustrated in Fig. 2.

The use of SIFT descriptors, spatial pyramidal pooling and SVM, have been shown to be crucial in producing leading performances for the BoW frameworks. The image representation begins with a low-dimensional pixel representation upon which local descriptors, such as SIFT [2], are extracted to create a robust and powerful representation. The local descriptors are sampled from the image using a dense, typically overlapping, sliding window over the image. A feature coding step maps each descriptor to a mid-level representation, and is currently the subject of much research that exploit machine learning techniques. Finally, spatial pyramids [4] with max-pooling [34] produce a single high-dimensional vector used to perform classification with linear SVMs. In the rest of this section, we enhance the feature coding step by introducing a deep hierarchical coding scheme based on restricted Boltzmann machines.

The main objective of this work is to encode powerful local descriptors, such as SIFT, using a deep hierarchy for image categorization. This is contrary to other deep networks [5], [9], [47], [48] that learn representations from pixels. While these attempts at learning from pixels have resulted in interesting representations, such as Garbor-like filters [29], [37], image categorization performances remain below those of the BoW framework. Our approach is to study the construction of a hierarchical coding scheme to learn from local descriptors as a starting point with a greater representational power than raw pixels. The introduction of hierarchical depth to the feature coding step is not a trivial one as it may not guarantee an improvement in categorization performances. We focus

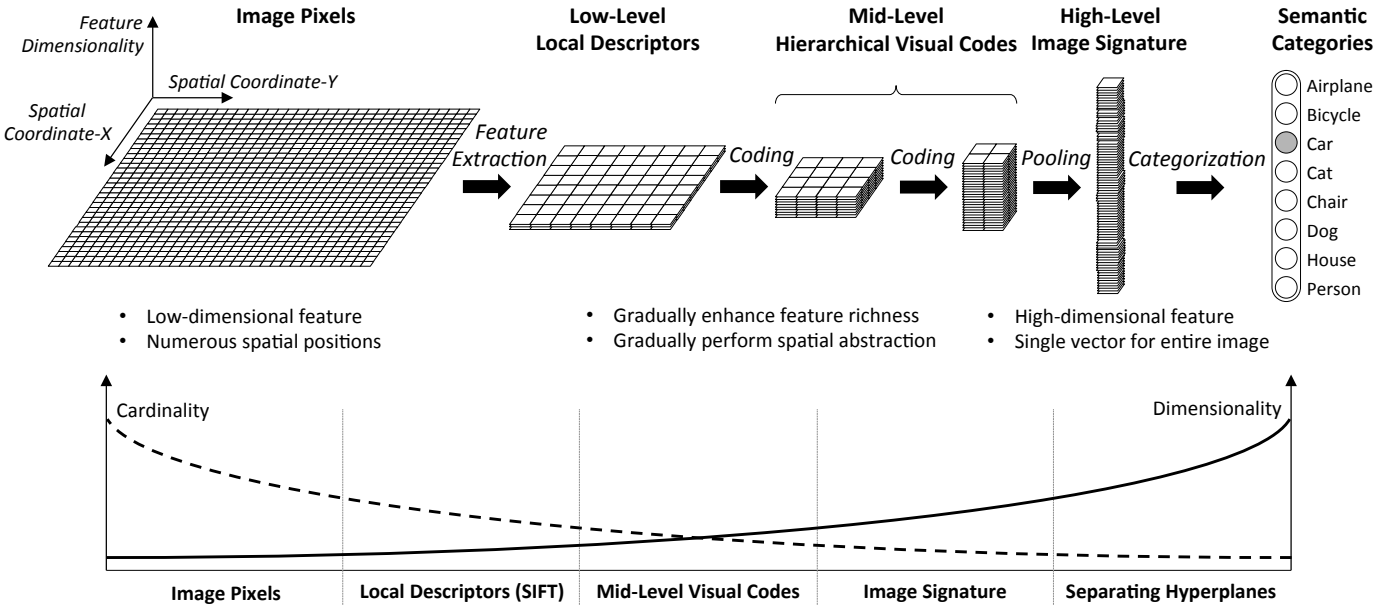


Fig. 2. The Bag of Words framework map representations from image pixels to low-level descriptors, mid-level visual codes and, eventually, to a high-level image signature, which is used for categorization. In the hierarchical architecture, the feature dimensionality increases progressively, enhancing the richness of the representation. Meanwhile, the features are gradually abstracted through the layers and the number of features (i.e. cardinality) over the image space is reduced. The graph illustrates the relation between dimensionality and cardinality of the representation through the various processes in the framework.

our approach on the following two aspects of deep feature coding: 1) encoding SIFT with a regularized shallow RBM network and 2) constructing the deep architecture to enhance modeling and representational strengths. This coding step allows for a gradual mapping of representation to achieve both feature enhancement and abstraction during the coding process (Fig. 2). To our knowledge this is the first work on learning deep representations from descriptors and we are able to outperform existing shallow feature coding methods and deep pixel-based methods (see Sec. V-B1).

A. Single-Layer Visual Architecture

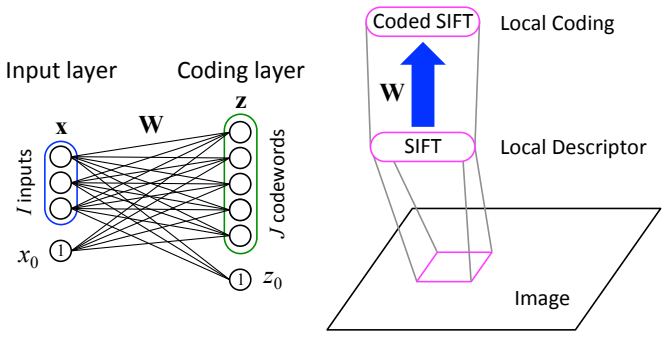
The basic component of the hierarchical architecture is the restricted Boltzmann machine (RBM) [19]. The RBM is a bipartite Markov random field that consists of an input feature layer $\mathbf{x}$ and coding layer $\mathbf{z}$ (Fig. 3(a)). The feature layer contains I dimensions corresponding to the dimensionality local image descriptor (i.e. 128 dimensions for SIFT). The coding layer has J latent units, each representing a visual codeword. Bias units, x_0 and z_0, are permanently set to one. The layers are fully-connected between them by an undirected weight matrix $\mathbf{W} \in \mathbb{R}^{I \times J}$.

Given a single input feature x and a set of weights $\mathbf{W}$, the coding z_j is computed as a feedforward encoding function $f_{enc}(\cdot, \cdot)$:

$$z_j = f_{enc}(\mathbf{x}, \mathbf{w}_j) = \sigma\left(\sum_{i=0}^{I} w_{ij} x_i\right), \tag{2}$$

where $\sigma(\cdot)$ is the sigmoid transfer function. To handle SIFT descriptors inputs, the descriptors are first ℓ_1-normalized so that each vector sums to a maximum of one, resulting in a quasi-binary input representation to suit the binary RBM.

Each RBM with 128 input units can be trained to model the SIFT descriptor (Fig. 3(b)). The SIFT encoding share the same weights that are tiled across the whole image. Already, the performance of this simple architecture is comparable to other leading methods (Sec. V-B1). It is also has faster inference than sparse-coding-based methods [10], [28], [34], [36], which require inference-time on reoptimization. With suitable regularization, RBMs can learn interesting representations from local gradient features [7] (see Sec. V-D1).



(a) An RBM (b) SIFT feature encoding

Fig. 3. (a) Structure of the restricted Boltzmann machine (b) A single RBM layer is used to encode a single SIFT descriptor in a shallow architecture.

B. Constructing a deep feature coding architecture

We now extend the single-layer architecture to a hierarchical one. In the case of deep fully-connected networks, RBM layers are stacked in a greedy layer-wise manner to form deep belief networks [12]. However, this conventional method of stacking fully-connected layers does not exploit spatial correlations and may not scale well to larger images. We found experimentally that for image classification tasks, directly stacking without taking into account spatial information does not yield desirable results.

To model the expanded spatial dimensionality in large images, we look to convolutional neural networks [9], [15], [16] and biologically inspired models [49], [50], [51], [52] for inspiration. These models incorporate spatial pooling as the architecture deepens so that the complexity of features increases, while the areas of representation over the original image are enlarged. When representing visual information, we often consider an object or a scene to be a composite of its sub-parts. When learning features hierarchies, it is sensible to perform spatial aggregation, whereby higher-level concepts are formed by abstracting over the lower-level ones.

The spatial aggregation operation correlates a spatial neighborhood of local codes or features, instead of treating them as being spatially independent. When this concatenation is performed at the descriptor level, this more complex feature becomes a *macro feature (MF)* [34]. At the higher-levels, spatial aggregation is done over the mid-level feature codes of the previous layer to create a *macro code*. The RBM models the joint interactions between the descriptive dimensions of its input layer and their spatial dependencies within the neighborhood. Spatial aggregation also helps the model adapt to large images through the abstraction over the image space.

In this work, we greedily stack one additional RBM with a new set of parameters $\mathbf{W}^{(2)}$ that aggregates within a neighborhood of outputs from the first RBM $\mathbf{W}^{(1)}$. The two-layer architecture for encoding macro features is shown in Fig. 4.

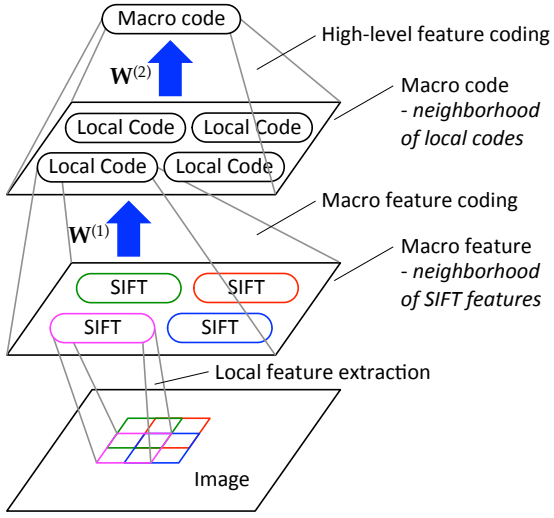


Fig. 4. A two-layer hierarchical architecture for coding SIFT-based macro features to local codes and subsequently to higher-level macro codes.

IV. Learning Visual Representations

Each layer in the architecture is trained as a restricted Boltzmann machine (RBM) [19] (see Fig. 3(a)). The joint configuration $(\mathbf{x}, \mathbf{z})$ of binary activation states in the both layers of the RBM has an energy given by:

$$E(\mathbf{x}, \mathbf{z}) = -\sum_{i=0}^{I}\sum_{j=0}^{J} x_i w_{ij} z_j. \tag{3}$$

The joint probability of states corresponds to:

$$P(\mathbf{x}, \mathbf{z}) = \frac{\exp\left(-E(\mathbf{x}, \mathbf{z})\right)}{\sum_{\mathbf{x},\mathbf{z}} \exp\left(-E(\mathbf{x}, \mathbf{z})\right)}\ . \tag{4}$$

The objective of the network is to maximize the probability of the input data x summed over all possible vectors in the latent layer z:

$$P(\mathbf{x}) = \frac{\sum_{\mathbf{z}} \exp\left(-E(\mathbf{x}, \mathbf{z})\right)}{\sum_{\mathbf{x},\mathbf{z}} \exp\left(-E(\mathbf{x}, \mathbf{z})\right)}\ . \tag{5}$$

Taking the gradient of the log probability yields

$$\frac{\partial \log P(\mathbf{x})}{\partial w_{ij}} = \langle x_i z_j \rangle_{data} - \langle x_i z_j \rangle_{model}\,, \tag{6}$$

where $\langle \cdot \rangle_{dist}$ denotes the expectation under the distribution $dist$. The first term consists of data driven activations that are clamped by the environment, while the model driven states are sampled from the stationary distribution of a free running network. However, finding $\langle x_i z_j \rangle_{model}$ is intractable as it requires performing infinite iterations of alternating Gibbs sampling via a symmetric decoder $f_{dec}(\cdot, \cdot)$:

$$x_i = f_{dec}(\mathbf{z}, \mathbf{w}_i) = \sigma\left(\sum_{j=0}^{J} w_{ij} z_j\right). \tag{7}$$

Hinton [53] proposed the contrastive divergence (CD) algorithm, that approximates the stationary distribution with a small finite number of sampling steps and used to update the weights by iterating over the training data:

$$\Delta w_{ij} = \varepsilon\left(\langle x_i z_j \rangle_{data} - \langle x_i z_j \rangle_{recon}\right), \tag{8}$$

where ε is a small learning rate. To avoid unnecessary sampling noise [54] and reduce the variance of the estimator [55], we employ Rao-Blackwellization [56].

A. Precise RBM Regularization

Using the maximum-likelihood as the only criteria to discover the mid-level representations might not be suitable for the image categorization task. Representations that are generatively learned with unlabeled image data can be regularized with inductive biases [57], which are the set of *a priori* assumptions about the nature of the target function. Imposing such prior structure can therefore be of great help for learning sensible representations for image categorization. This forms the motivation behind regularizing RBM learning with an additional term $h(\mathbf{z})$ weighted by λ to the optimization problem:

$$\arg\min_{\mathbf{W}} \quad -\sum_{k=1}^{K} \log\left(\sum_{\mathbf{z}} P(\mathbf{x}_k, \mathbf{z}_k)\right) + \lambda h(\mathbf{z}), \tag{9}$$

performed over a training set of K examples.

We previously discovered that the RBM can be precisely regularized to influence the activation of individual codewords j in response to each input example k by introducing specific target activations $p_{jk} \in [0, 1]$ [37]. The targets can be organized into a matrix $\mathbf{P} \in \mathbb{R}^{J \times K}$, with each row $\mathbf{p}_j$ representing the desired activation vector of codeword z_j with respect to the set of K input features, while each column $\mathbf{p}_k$ denotes the population code given input k. To regularize the RBM, $h(\mathbf{z})$ can be defined using the cross-entropy loss:

$$h(\mathbf{z}) = -\sum_{j=1}^{J}\sum_{k=1}^{K} p_{jk} \log z_{jk} + (1 - p_{jk}) \log(1 - z_{jk}). \tag{10}$$

This essentially matches data-sampled activations to target activations, while maximizing the likelihood of the data. We can further perceive $x_{ik,target} := x_{ik}$ and $z_{jk,target} := p_{jk}$ as a sampling from the input and latent target distributions respectively. After fusing the batch-averaged gradient of the regularizer, the learning rule of Eq. 8 can now be modified as

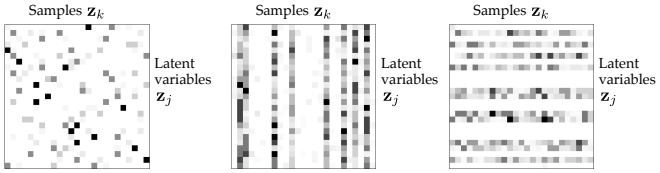
$$\Delta w_{ij} = \gamma \langle x_i z_j \rangle_{data} + \eta \langle x_i z_j \rangle_{target} - \varepsilon \langle x_i z_j \rangle_{recon}\,, \tag{11}$$

where η is the learning rate of the target distribution and $\gamma = \varepsilon - \eta$ is the modified learning rate of the data distribution due to the regularization. The derivation of the update rule is described in the Appendix.

B. Sparse and Selective Regularization

From the regularization method presented in Section IV-A, we can bias the representations to suit visual coding by introducing selectivity and sparsity [58]. These coding properties improve visual information efficiency, memory storage capacity and pattern discrimination [59], [60]. Sparsity is a statistic of the *population* of codewords in response to a *single input example*. On the other hand, selectivity describes the activation of *one visual codeword* across a set of examples.

Besides having the same mean values, selectivity and sparsity are uncorrelated [61]. If only a small proportion of codewords respond to the inputs, the codes are always sparse, but none of them are selective, making feature discrimination difficult (Fig. 5(c)). When all codewords are selective to the same feature instances, sparse codes will not be produced and discrimination also suffers (Fig. 5(b)). A graphical explanation of this phenomenon is shown in Fig. 5.



(a) Sparse and selective (b) Only selective (c) Only sparse

Fig. 5. Comparing different coding strategies with the activation matrix $\mathbf{Z}$. (a) A well trained dictionary exhibits both selectivity and sparsity. Activations have high variance and are uncorrelated. (b) Selective-only coding may cause many features to be ignored or over represented. (c) Sparse-only coding alone might lead to codewords that have strong correlation or are silent.

Some existing work coarsely use the mean activity of a codeword across examples to regularize RBMs [29], [62]:

$$h(\mathbf{z}) = \sum_{j=1}^{J} \left\| \hat{p} - \frac{1}{K} \sum_{k=1}^{K} z_{jk} \right\|^2, \tag{12}$$

where $\hat{p}$ is the desired (usually low) mean codeword activation. However, each codeword is penalized using a single global statistic without regard of its selectivity to individual examples, so the condition could be satisfied even when the codeword is not selective. Additionally, sparsity is not considered. This contrasts with methods only encouraging representations to be sparse [10], [25], [28].[1]

Our aim is to map an RBM's input to a latent space where the variations between input examples are retained, while encouraging individual latent dimensions to specialize. We employ the precise RBM regularization (Sec. IV-A) to induce *both* sparsity and selectivity, by biasing data-sampled activations $\mathbf{Z} \in \mathbb{R}^{J \times K}$ to match appropriate targets $\mathbf{P}$. The activations across the rows and columns of the $\mathbf{P}$ are fitted to distributions rather than being summarized by their averages. For every row and column of data-sampled responses $\mathbf{z} \in \mathbb{R}^N$ (Fig. 6(a)), we transform the activations to fit a positively skewed long-tail target distribution (Fig. 6(c)):

$$p_n = (rank(z_n, \mathbf{z}))^{\frac{1}{\mu}-1}, \tag{13}$$

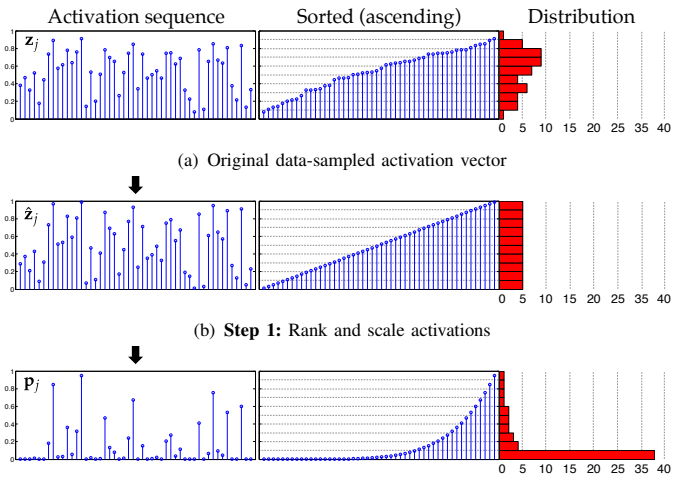
where $rank(z_n, \mathbf{z})$ performs histogram equalization by sorting the activations and assigning a value between 0 to 1 based on the rank of z_n in $\mathbf{z}$, with smallest given a value of 0 and the largest with 1. The target mean $\mu \in (0, 1)$ positively skews the distribution if $\mu < 0.5$.

To obtain targets $\mathbf{P}$ that are jointly sparse and selective, we first map all J rows of K-element vectors to their sparsity target distribution, followed by all columns for selectivity [7], [37], [65]. The additional training cost is the time needed to sort the activation matrix in both dimensions, which is $J \log J + K \log K$. With a distribution based target, some activations will be pulled up while most will be encouraged to be lower. This is as opposed to penalizing activations equally with a mean-based regularization [29], [62].

Our preliminary study using this regularization scheme yields positive results for learning shallow visual dictionaries [7]. We now use it to regularize each layer of representation in our deep architecture using the same distribution-based technique. The lower-level dictionary models the data in the same manner as a shallow dictionary [7], while the higher-level visual dictionary seeks to achieve a sparse and selective representation of concatenated mid-level codes (see Sec. III-B).

C. Supervised Fine-Tuning

So far, the training algorithms have focused on unsupervised learning, yet it is also important for supervised learning to be introduced. Using top-down discrimination to model the data is an effective way to approach a classification task. This is relatively simple for shallow dictionaries, as shown in our preliminary study [7], by using the error backpropagation algorithm [20], [21]. However, in a deep architecture, the discriminative backpropagated signal diffuses as it travels down the network, making the optimization problem more challenging. Similarly, the importance of supervised learning increases as we stack layers, since the quality of the generative representations for categorization diminishes at the upper layers (empirically shown in Table II). A combination of both bottom-up and top-down learning needs to be performed to train an effective network. Supervision can be introduced either concurrently as a hybrid model [66], or through a separate fine-tuning phase [12]. In this work, after greedily stacking two RBMs, the parameters are first fine-tuned through a series of two steps, first using a combination of bottom-up and top-down signals [67] and later using error backpropagation.



(a) Original data-sampled activation vector

(b) **Step 1:** Rank and scale activations

(c) **Step 2:** Transform vector to fit a long-tailed distribution

Fig. 6. A succession of two steps transforms a set of data-sampled activations to their targets. For illustrative purposes, the sequence of latent activations on the left is sorted in an ascending order of their activation level (middle) and its histogram is displayed on the right. (a) The original activation sequence may take the form of any empirical distribution. (b) Step 1 ranks the signals and scales them between 0 and 1, resulting in a uniform distribution within that interval. (c) Step 2 maps the ranked signals to fit a predefined long-tailed distribution to obtain $\mathbf{p}$. Only few target activations are encouraged to be high, while most are low.

In the first step, a new "classifier" RBM, with weights $\mathbf{W}^{(3)}$, connects the outputs of the second RBM to an output layer $\mathbf{y} \in \mathbb{R}^C$, with each unit corresponding to a class label c. This RBM is trained by directly associating the $\mathbf{z}^{(2)}$ to target outputs $\mathbf{y}$:

$$\Delta w_{ic}^{(3)} = \varepsilon(\langle z_i^{(2)} y_c \rangle_{data} - \langle z_i^{(2)} y_c \rangle_{recon}). \tag{14}$$

Next, all the layers are bound together by using top-down sampled signals as targets for bottom-up activations [67]. An initial up-pass generates unbiased samples from the bottom-up. Starting from the top-most layer, a down-pass then samples target activations $\mathbf{z}^{(l)}$ using the biased activations of the layer $(l+1)$ above, as follows:

$$\tilde{z}_{j,target}^{(l)} = f_{dec}(\phi^{(l+1)} \tilde{\mathbf{z}}_{target}^{(l+1)} + (1 - \phi^{(l+1)}) \mathbf{z}_{data}^{(l+1)}, \mathbf{w}_j^{(l+1)}), \tag{15}$$

with $\phi^{(l)}$ being a hyperparameter. Additionally, alternate Gibbs sampling chains are formed from each paring of layers based

[1] Besides sparsity and selectivity, topographically-organized regularization [47], [63], [64] is also a possible (see Goh et al. [38]).

on the "up"-pass activations. The update equation for weights of layer l are defined as follows:

$$\Delta w_{ij}^{(l)} = \gamma \langle z_i^{(l-1)} z_j^{(l)} \rangle_{data} + \eta \langle z_{i,data}^{(l-1)} \tilde{z}_{j,target}^{(l)} \rangle - \varepsilon \langle z_i^{(l-1)} z_j^{(l)} \rangle_{recon}, \quad (16)$$

where $\mathbf{z}^{(0)} = \mathbf{x}$ for the first RBM, and $\mathbf{z}_{data}^{(3)} = \mathbf{z}_{target}^{(3)} = \mathbf{y}$ such that the topmost RBM update follows Eq. 14. All RBMs in the architecture are updated concurrently. This step fine-tunes the existing hierarchical visual dictionary by introducing intermediate learning signals between unsupervised learning and highly discriminative learning in the next step.

In the second visual dictionary fine-tuning step, we used the discriminative softmax cross-entropy loss to penalize feature-based classification errors. The errors are then backpropagated through the three sets of parameters consisting of two layers of visual dictionaries and one layer of feature-level classifier.

Overall, the training of parameters for the entire BoW model can be seen in 6 distinct steps, grouped into 3 phases (Fig. 7). The first phase performs greedy layer-wise unsupervised learning of the two levels of RBM visual dictionaries from the bottom-up. The second phase uses top-down sampled activations to fine-tune bottom-up learning. Finally, the third phase further fine-tunes the visual dictionaries with error back-propagation and learns a discriminative SVM classifier on the image-level representations. Inference is a simple feedforward pass though the BoW pipeline to obtain the category label.

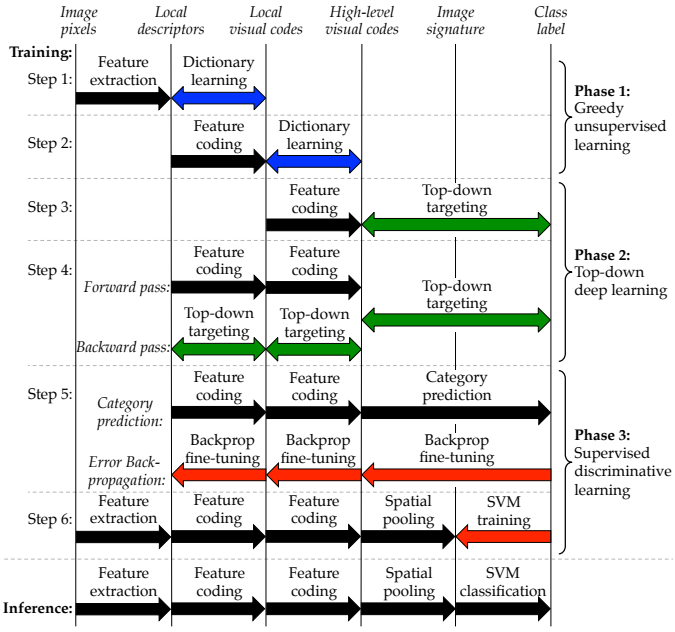


Fig. 7. Overall, the parameters of the BoW model are optimized in six steps. The steps are grouped into three phases: 1) greedy supervised learning, 2) top-down regularized deeps learning and 3) supervised discriminative learning. Inference consists of a single feedforward pass through the BoW pipeline.

V. Experiment Results and Discussion

A. Experimental Setup

We performed experimental evaluations on object and scene categorization tasks using the unsupervised and supervised variants of the three architectures (Section III) with different depths (shallow or deep) and input features used (SIFT or macro features (MF)), as illustrated in Fig. 3(b) and Fig. 4.

1) Image Datasets: We used three datasets for our evaluations - Caltech-101, Caltech-256 and 15-Scenes. The Caltech-101 dataset [68] contains 9,144 images belonging to 101 object categories and one background class. There are between 31 to 800 images in each category. The Caltech-256 dataset [69] extends the original Caltech-101 dataset to 256 object classes and 30,607 images. The 15-Scenes dataset [4] consists of 4,485 images from 15 different scene categories.

2) Local Feature Extraction: Before feature extraction, images were resized to fit within a 300×300 pixel box, retaining their original aspect ratios. SIFT descriptors scaled at 16×16 pixels were densely sampled from each image at 8 pixel intervals. Macro features were pooled from 2×2 neighborhoods of SIFTs extracted at 4 pixel intervals. This setup follows that of existing BoW approaches [34]. The SIFT descriptors were ℓ_1-normalized by constraining each vector to sum to a maximum of one.

3) Dictionary Learning and Evaluation Setup: A set of 200,000 randomly selected feature examples (descriptors or codes) were used as the training set for unsupervised dictionary learning for each layer. In experimental trial, a number of the training images per category (15 or 30 for Caltech-101; 30 or 60 for Caltech-256; 100 for 15-Scenes) were randomly drawn for supervised fine-tuning. The same train-test split was also used for training the SVM classifier in the final stage of the BoW framework. The remaining images that were not used for training were then used for testing and the mean class-wise accuracies averaged over 10 trials was reported.[2]

4) Spatial Pyramids and SVM Classification: From the visual codes produced by each architecture, a three-level spatial pyramid [4] with max-pooling [34] is used to form the final image representation based-on the typical pooling grids of 4×4, 2×2 and 1×1. Finally, we trained a linear SVM to perform multi-class classification of the test images.

B. Image Categorization Performance

1) Comparison with Other Methods: At the bottom of Table II, we present the image categorization results on the Caltech-101 and 15-Scenes datasets using the different setups of our architecture – using different input features and architecture depth, as well as with and without supervised fine-tuning. Our best image categorization results obtained on the Caltech-101 dataset were $72.1 \pm 1.3\%$ using 15 training images and $79.7 \pm 0.9\%$ with 30 training images. For the 15-Scenes dataset, we obtained a classification accuracy of $86.4 \pm 0.6\%$. The best architecture was consistently the deep supervised dictionary trained on macro feature inputs. These are competitive results for BoW methods focusing on feature coding with standard pooling setups. Moreover, performance was consistently good across both datasets. We also observe that macro features consistently outperform SIFT descriptors,

[2]We followed the standard evaluation metric by using all categories (including the background) for both training and evaluation. http://www.vision.caltech.edu/Image_Datasets/Caltech101

by about 3% on the Caltech-101 dataset and 1.5% for the 15-Scenes dataset. This difference in performance validates the results reported by Boureau et al. [34].

In Table II, we compare our results with other feature coding strategies that follow the same BoW pipeline using only a single feature type and standard pooling methods. We are favorably positioned within both unsupervised and supervised methods. Compared against all other coding-focused schemes, we achieved the leading results with the supervised deep architecture. At this juncture, RBM-based method for visual dictionary learning appear to gain a slight edge over methods with non-learned assignment coding or sparse coding. RBM-based methods also have faster inference speeds as compared to sparse coding methods [10], [34], [36] and a more compact representation than some coding strategies [28].

Table III presents a comparison with other methods that focus on other (non-feature-coding) aspects of image categorization, such as image modeling. Our architecture significantly outperforms all recent pixel-based convolutional methods [9], [47], [72]. The main difference of our hybrid deep architecture from these methods is that we exploit the BoW framework, particularly the SIFT descriptors, spatial pyramidal pooling and SVM classification. This is particularly useful when the datasets do not have many labelled training examples to learn from. Additionally, we also utilize supervision during training.

Amongst the other post-feature-coding (pooling and classification) methods, the methods by Duchenne et al. [73] and Feng et al. [74] reported impressive performances on the Caltech-101 dataset, which are currently the best accuracies on the dataset. Duchenne et al. [73] used graph matching to encode the spatial information of representations learned by sparse coding [34]. Feng et al. [74] build upon LLC sparse codes [36] to perform pooling in a discriminative manner, using an ℓ_p norm aggregation strategy to pool codes in between sum and max, and combined with a spatial weighting term optimized for classification. These methods [73], [74] address the image categorization problem in a completely different direction as we do. Our deep feature coding method, being easily incorporated in the BoW framework, may be complementarily combined with these methods to possibly boost performances.

2) Discussion on Depth and Supervision: The deep supervised architecture produced the best performing visual dictionary. We discuss the empirical performance of both depth and supervision, and offer possible explanations of the results.

We begin with the basic shallow unsupervised architecture, which already bring us close to the state-of-the-art results. When we stack an additional unsupervised layer, we observe a consistent fall in the performance. We think that this may be due to the model deviating from the classification objective as layers are added – a problem that may exist even with superior generative learning on the maximum likelihood criterion. It may not be sensible to increase the architecture's depth if we are unable to adapt the entire set of parameters to suit the eventual image categorization task.

As shown in Table II, classification results improve when supervised fine-tuning is performed on both the shallow and

TABLE II
PERFORMANCE COMPARISON WITH BOW FEATURE CODING METHODS

Feature Coding Method	Codebook Size	Caltech-101		15-Scenes
		15 tr.	30 tr.	100 tr.
Non-Learned Coding				
Hard assignment [4]	200	56.4	64.6 ± 0.8	81.1 ± 0.3
Kernel codebooks [27]	200	-	64.1 ± 1.5	76.7 ± 0.4
Soft assignment [70]	1000	-	74.2 ± 0.8	82.7 ± 0.4
Sparse Dictionary Learning and Coding				
ScSPM [10]	1024	67.0 ± 0.5	73.2 ± 0.5	80.3 ± 0.9
LLC [36]	2048	65.4	73.4	-
Sparse coding & max-pooling [34]	1024	-	75.7 ± 1.1	84.3 ± 0.5
Sparse spatial coding [71]	1024	69.98 ± 0.9	77.59 ± 0.5	-
Multi-way local pooling [28]	1024×65	-	77.3 ± 0.6	83.1 ± 0.7
Restricted Boltzmann Machine				
Sparse RBM [8]	4096	68.6	74.9	-
CRBM [8]	4096	71.3	77.8	-
Supervised Dictionary Learning				
Discriminative sparse coding [34]	2048	-	-	85.6 ± 0.2
LC-KSVD [31]	1024	67.7	73.6	-
Our Architectures (SIFT Feature)				
Unsupervised Shallow	2048	66.5 ± 1.6	74.7 ± 1.1	84.2 ± 0.9
Supervised Shallow	2048	67.6 ± 1.2	75.6 ± 1.0	84.3 ± 0.7
Unsupervised Deep	2048	62.5 ± 1.4	69.9 ± 1.2	79.6 ± 0.5
Supervised Deep	2048	69.3 ± 1.1	77.2 ± 0.8	85.2 ± 0.5
Our Architectures (Macro Feature)				
Unsupervised Shallow	2048	70.0 ± 1.9	78.0 ± 1.4	85.2 ± 0.6
Supervised Shallow	2048	71.1 ± 1.4	79.1 ± 1.3	85.6 ± 0.5
Unsupervised Deep	2048	65.3 ± 1.5	72.8 ± 1.1	82.5 ± 0.6
Supervised Deep	2048	$\mathbf{72.1 \pm 1.3}$	$\mathbf{79.7 \pm 0.9}$	$\mathbf{86.4 \pm 0.6}$

TABLE III
COMPARISON WITH NON-FEATURE-CODING METHODS

Method	Caltech-101	
	15 tr.	30 tr.
Our Architecture		
Supervised Deep (Macro Feature)	72.1 ± 1.3	79.7 ± 0.9
Recent Convolutional Networks		
Convolutional Deep Belief Net [9]	57.7 ± 1.5	65.4 ± 0.5
Convolutional Neural Network [47]	-	66.3 ± 1.5
Deconvolutional Network [72]	58.6 ± 0.7	66.9 ± 1.1
Hierarchical Sparse Coding [48]	-	74.0 ± 1.5
Post-Feature-Coding Methods		
NBNN [75]	65.0 ± 1.1	70.4
NBNN kernel [76]	69.2 ± 0.9	75.2 ± 1.2
Graph-matching kernel [73]	75.3 ± 0.7	80.3 ± 1.2
GLP [74]	70.3	82.6
SLC [45]	72.7 ± 0.2	81.0 ± 0.2

deep architectures. There is only a slight improvement when supervision is added to shallow architectures. However, the gains are particularly large for deep architectures, so much so that it overcomes the deficit of performance between the shallow and deep unsupervised architectures. This is perhaps due to the deep architecture's intrinsic capacity to encode more complex representations within the structure, that increases the chances of class-wise separability. It shows the importance of supervised fine-tuning, especially for an architecture that is several layers deep. Ultimately, it was the combination of supervised fine-tuning, architecture depth and macro features that delivered the best image categorization scores.

3) Computational Resources: Due to the sparse and selective regularization, the codewords learned encode generic image structure and tend to be very diverse, resulting in concise codebooks with few codewords. As a result, our method remains very competitive even as the codebook size is reduced. As compared to the best performing method of sparse coding [28], our final image signature is 32.5 times smaller. We also use half the number of codeword as compared to the best RBM-based approach [8]. In both cases, we outperform the methods in terms of classification performance.

Feature coding is fast during inference because we exploit the encoder nature of the trained RBM network. The inference time is the same, whether supervised or unsupervised, because we merely perform a two-layer feedforward computation for each feature to obtain its coding. When implemented, descriptors can be computed concurrently in batches. The advantage of inference speed is especially significant when compared against sparse coding methods [10], [34], [36], which have to run the sparse optimization during inference. Experimentally, we record an inference speedup of 80 times, relative to the ScSPM method [10].

C. Shallow and Deep Transfer Learning

1) Between Caltech-101 and Caltech-256: Using a model with the same complexity as that for the Caltech-101 dataset, we achieved competitive image categorization performances on the Caltech-256 dataset, with mean accuracies of 41.5 ± 0.7 and 47.2 ± 0.9, using 30 and 60 training examples respectively. Due to the similarity between Caltech-101 and Caltech-256, we attempted to transfer the unsupervised dictionaries learned using Caltech-101 to classify Caltech-256 images, in the spirit of self-taught learning [77]. Assuming that unsupervised dictionary learning had been done using the Caltech-101 dataset, transferring the learned representation reduces the amount of training time required for the entire deep model Caltech-256 dataset. The results of this study together with results for other competitive methods are presented in Table IV. The Caltech-256 dataset is used for supervised fine-tuning where reported.

We found that with a shallow architecture, the unsupervised dictionary from Caltech-101 essentially performed the same as that trained from Caltech-256. However, when we stack a second layer trained using the Caltech-101 datatset, the errors resulting from the transfer of dictionary become pronounced. If the first layer is trained with the Caltech-101 dataset, while the second layer is trained on the Caltech-256 dataset, the results is again no different from when both layers are trained with the Caltech-256 dataset. From this, we believe that the first layer is able to models generic spatially-local dependencies. As we

TABLE IV
PERFORMANCE ON CALTECH-256 FOR SHALLOW AND DEEP TRANSFER LEARNING SETUPS

Method	Training set(s)		30 tr.		60 tr.	
	Layer 1	Layer 2	Unsupervised	Fine-tuned	Unsupervised	Fine-tuned
Our Architectures - Standard Shallow & Deep Learning Setups						
Shallow	Caltech-256	-	41.0 ± 1.0	41.1 ± 0.8	46.1 ± 0.9	46.0 ± 0.8
Deep	Caltech-256	Caltech-256	38.9 ± 0.8	41.5 ± 0.7	44.7 ± 0.8	47.2 ± 0.9
Our Architectures - Transfer Learning Setups						
Shallow	Caltech-101	-	40.8 ± 1.1	41.0 ± 1.0	45.8 ± 0.9	45.9 ± 1.0
Deep	Caltech-101	Caltech-101	36.5 ± 1.0	38.3 ± 0.9	41.4 ± 1.0	44.2 ± 1.0
Deep	Caltech-101	Caltech-256	39.6 ± 0.9	41.7 ± 0.9	44.0 ± 1.1	47.0 ± 1.0
Other Competitive Methods						
ScSPM [10]	Caltech-256		34.0		40.1	
Graph-matching kernel [73]	Caltech-256		38.1 ± 0.6		-	
LLC [36]	Caltech-256		41.2		47.7	
CRBM [8]	Caltech-256		42.1		47.9	
GLP [74]	Caltech-256		43.2		-	

spatially aggregate while stacking layers, the representation becomes more category specific and increase in semantic value. So, transferring higher-level information from Caltech-101 deeper into the architecture will not be as useful as learning directly from the Caltech-256 dataset itself.

2) Between Caltech-101 and 15-Scenes: The same experiment was also performed to study the possibility of transferring learned representations between datasets of different natures, objects (Caltech-101) and scenes (15-Scenes) in this case. The experimental results of the transferring shallow and deep representations between the two datasets are presented in Fig. 8, where the labelled data for supervised learning is obtained from the respective test dataset. The general observation is similar the previous one between Caltech-101 and Caltech-256, that the deeper the trained representation, the harder to adapt to another another dataset.

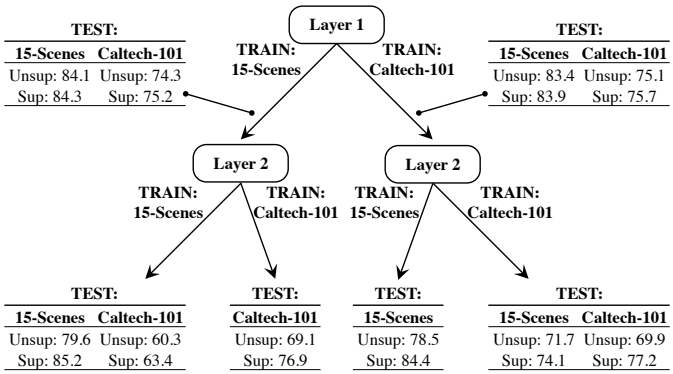


Fig. 8. Transfer learning results between Caltech-101 and 15-Scenes. Transfer learning works well when the representation transferred is shallow.

D. Analysis of Visual Dictionary Learning

1) Codeword Visualization: We visualized the codewords trained on SIFT descriptors extracted from the Caltech-101 dataset [68]. Each codeword is extracted as a filter over the 128-dimension SIFT feature space. Each SIFT descriptor encodes quantized gradients (8 bins) within a 4×4 neighborhood grid. We split the filter into the same spatial grid and assign to the grid the dominant orientation of the partition, with a strength proportional to the reconstructed response of the filter. We assign each of the eight orientations a distinctive hue, while the intensity shows the local response strength of a partition. The result of the visualization is shown in Fig. 9(a), where each square represents a codeword. It is interesting to observe that the RBM automatically discovers coherent structure. For many codewords, opposing gradients are paired and have consistent directions. For example, red-cyan pairings tend to occur left and right of each other. A further analysis of the visualization is described in Fig. 9(b), 9(c), 9(d) and 9(e). The diversity between the codewords leads to differentiation and discrimination between features in the coding layer. We found the codewords learned from SIFT extracted from the Caltech-101, Caltech-256 and 15-Scenes datasets to be visually similar. This leads to the potential that the shallow dictionaries learned are generic enough to be transferred between datasets (Section V-C).

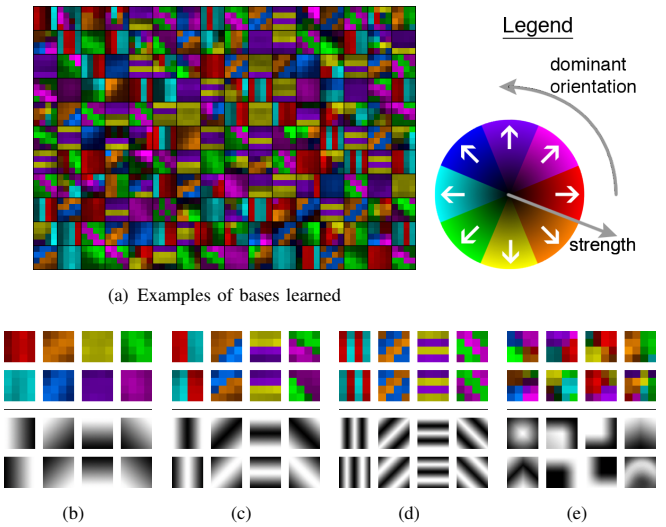


Fig. 9. (a) Visualization of visual dictionary learned from SIFT descriptors by a shallow architecture. The codewords are observed to encode a variety of image structure, such as (b) smooth gradients, (c) lines and edges, (d) textured gratings or (e) other complex features, such as corners, bends and center-surround features. Hypothetical pixel intensities leading to strong responses for the codewords are shown below each set of codeword examples.

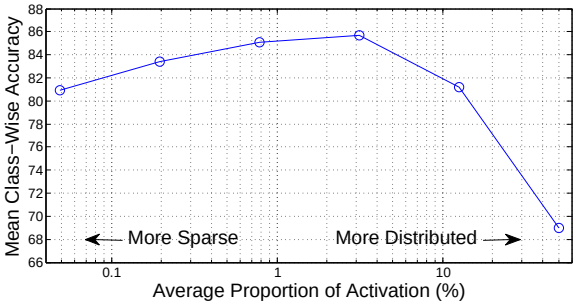


Fig. 10. Results on 15-Scenes with unsupervised RBM learning jointly regularized with sparsity and selectivity. Performance degrades as the coding gets too sparse or too distributed.

2) Effects of Sparse and Selective Regularization: The average selectivity of a population of codewords is equivalent to the sparsity of the population averaged across examples. We analyze a metric relative to the size of the visual dictionary. In Fig. 10, we observe the effects of varying levels of induced selectivity and sparsity on image categorization with 15-Scenes dataset using the shallow unsupervised architecture trained on macro features. The performance suffers on both ends of the spectrum when the representation is too sparse and selective, or too distributed and broady tuned.

We note the importance of the joint sparse and selective regularization. The results on the Caltech-101 dataset (15 training examples), using a shallow distributed RBM (51.7%) and RBMs regularized with sparse long-tailed distributions (45.5%) and selective long-tailed distributions (36.8%), were significantly lower compared to the jointy sparse and selective RBM (66.8%). Another way to analyze the effects of sparsity and selectivity is to perform image categorization with varying levels of induced sparsity and selectivity (Fig. 11). The image categorization results peak when there is a suitably balanced

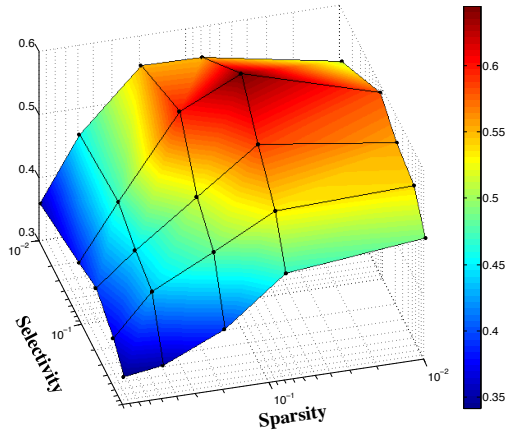


Fig. 11. Relative contributions of sparsity and selectivity on categorization results on Caltech-101 (15 training examples) using a shallow architecture.

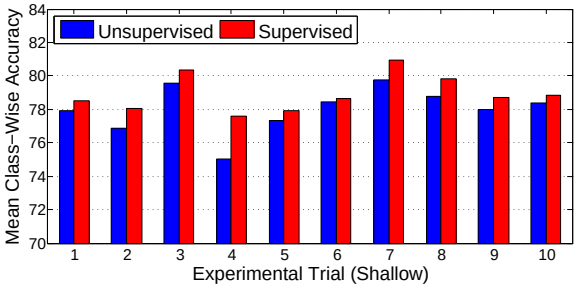
amount of sparsity and selectivity induced. This highlights the need for a jointly sparse and selective regularizer. In general, categorization results tend to peak with μ set in the range of 0.02 to 0.1 across the datasets.

3) Impact of Supervised Fine-Tuning: As observed in Table II, supervised fine-tuning improves the classification accuracy, slightly in the shallow model and more significantly in deep architectures. It is obvious that the gains due to supervision are statistically significant for the deep architecture, however its benefits on the shallow architecture remains uncertain. An analysis of individual experimental trials with 30 training examples on Caltech-101 (Fig. 12) reveals that results of every trial is improved through fine-tuning. The average improvement per trial for the shallow and deep architectures are $0.9 \pm 0.6\%$ and $6.8 \pm 0.6\%$ respectively. Statistically, the gain is always positive for both architectures, but supervision is especially important to empower deep visual dictionaries.

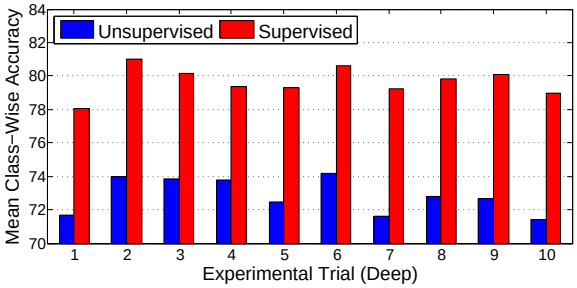
VI. Conclusions

In this paper, we presented a deep hierarchical architecture to encode SIFT descriptors in the BoW framework. The problem was approached in two main directions: 1) unsupervised learning of RBM modules regularized to be jointly sparse and selective, and 2) stacking these building blocks while capturing spatial correlations. These RBMs were later fine-tuned using top-down supervision – an empirically important step for deep architectures. The result is a four-layer BoW architecture consisting of an input layer of SIFT features, an output layer of category labels and two intermediate layers of spatially aggregating and supervised fine-tuned representations. This design achieved competitive results among the feature-coding family of methods, on both the Caltech-101 and 15-Scenes image categorization datasets.

The integration of deep learning and the BoW model, being in its inceptive phase, is open to many propositions of future studies. For our part, we hope to bridge the gap for optimizing local representations using global image labels. We are also currently studying techniques to better integrate bottom-up with top-down signals for computer vision problems.



(a) Individual trials for the shallow architecture



(b) Individual trials for the deep architecture

Fig. 12. Results for 10 trials on Caltech-101 (30 examples) for the (a) shallow and (b) deep architectures using macro features. When supervised fine-tuning improves the results for every trial. The performance boost is substantial for the deep model.

Appendix
Derivation of Regularized Update Rules

This appendix details the steps to derive the RBM update rules for the unsupervised phase consisting of the combination of maximum likelihood approximation and precise regularization. We begin with Eq. 9 and 10, where following optimization problem was posed:

$$\arg\min_{\mathbf{W}} \quad -\sum_{k=1}^{K} \log \sum_{\mathbf{z}} P(\mathbf{x}_k, \mathbf{z}_k) + \lambda \sum_{j=1}^{J} \sum_{k=1}^{K} \mathcal{L}(z_{jk}, p_{jk}), \tag{17}$$

where λ is a regularization constant and $\mathcal{L}(z_{jk}, p_{jk})$ is simply the cross-entropy loss between the data-sampled activation z_{jk} and the target activation p_{jk}:

$$\mathcal{L}(z_{jk}, p_{jk}) = -p_{jk} \log z_{jk} - (1 - p_{jk}) \log(1 - z_{jk}). \tag{18}$$

Let the total input for z_{jk} be $u_{jk} = \sum_{i=0}^{I} w_{ij} x_{ik}$. We take the partial derivative of $\mathcal{L}(z_{jk}, p_{jk})$ with respect to w_{ij} and apply a negative constant $-\eta \propto \lambda$ to reverse the error of z_{jk}:

$$\begin{aligned} \Delta w_{ij}(k) &= -\eta \frac{\partial \mathcal{L}}{\partial w_{ij}} = -\eta \frac{\partial \mathcal{L}}{\partial z_{jk}} \frac{\partial z_{jk}}{\partial u_{jk}} \frac{\partial u_{jk}}{\partial w_{ij}} \\ &= -\eta \left(\frac{1 - p_{jk}}{1 - z_{jk}} - \frac{p_{jk}}{z_{jk}} \right) \left(z_{jk} \right) \left(1 - z_{jk} \right) x_{ik} \\ &= -\eta \, x_{ik} (z_{jk} - p_{jk}) \end{aligned} \tag{19}$$

Together with the contrastive divergence gradient, the batch-wise parameter update is:

$$\begin{aligned} \Delta w_{ij} &= \varepsilon (\langle x_i z_j \rangle_{data} - \langle x_i z_j \rangle_{recon}) - \eta \, \langle x_{i,data} (z_{j,data} - p_j) \rangle \\ &= \gamma \, \langle x_i z_j \rangle_{data} + \eta \, \langle x_i z_j \rangle_{target} - \varepsilon \, \langle x_i z_j \rangle_{recon} \end{aligned} \tag{20}$$

where $\gamma = \varepsilon - \eta$ is the new learning rate for the data distribution as modified by the target distribution and $\langle \cdot \rangle$ denotes the an averaging over the K training samples in each batch and x_{ik} and z_{jk} are sampled from the distribution denoted by the subscript. Here, sampling from the target input distribution is equivalent to the data-sampled input themselves (i.e. $x_{i,target} := x_{i,data}$), while sampling from the latent target distribution $z_{j,target}$ directly yields p_j.

REFERENCES

[1] J. Sivic and A. Zisserman, "Video Google: A text retrieval approach to object matching in videos," in *ICCV*, 2003.

[2] D. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, pp. 91–110, 2004.

[3] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *CVPR*, 2005.

[4] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *CVPR*, 2006.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *CVPR*, 2009.

[7] H. Goh, N. Thome, M. Cord, and J.-H. Lim, "Unsupervised and supervised visual codes with restricted boltzmann machines," in *ECCV*, 2012.

[8] K. Sohn, D. Y. Jung, H. Lee, and A. Hero III, "Efficient learning of sparse, distributed, convolutional feature representations for object recognition," in *ICCV*, 2011.

[9] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *ICML*, 2009.

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

[11] Y. Bengio, "Learning deep architectures for AI," *Foundations and Trends in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[12] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief networks," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.

[13] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *NIPS*, 2006.

[14] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, no. 2, pp. 251–257, 1991.

[15] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541 – 551, 1989.

[16] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *P. IEEE*, vol. 86, no. 11, pp. 2278–2324, November 1998.

[17] M. Ranzato, F. Huang, Y. Boureau, and Y. LeCun, "Unsupervised learning of invariant feature hierarchies with applications to object recognition," in *CVPR*, 2007.

[18] G. E. Hinton, "To recognize shapes, first learn to generate images," in *Computational Neuroscience: Theoretical Insights into Brain Function*, P. Cisek, T. Drew, and J. Kalaska, Eds. Elsevier, 2007.

[19] P. Smolensky, "Information processing in dynamical systems: Foundations of harmony theory," in *Parallel Distributed Processing: Vol. 1: Foundations*, D. E. Rumelhart, J. L. McClelland, and the PDP Research Group, Eds. Cambridge: MIT Press, 1986, pp. 194–281.

[20] Y. LeCun, "Une procédure d'apprentissage pour réseau a seuil asymmetrique (a learning scheme for asymmetric threshold networks)," in *Cognitiva 85*, Paris, France, 1985, pp. 599–604.

[21] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533 – 536, October 1986.

[22] G. E. Hinton and R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, pp. 504–507, 2006.

[23] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *ICML*, 2008.

[24] B. A. Olshausen and D. J. Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature*, vol. 381, no. 6583, p. 607, 609 1996.

[25] K. Kavukcuoglu, P. Sermanet, Y. Boureau, K. Gregor, M. Mathieu, and Y. LeCun, "Learning convolutional feature hierachies for visual recognition," in *NIPS*, 2010.

[26] L. Bo and X. R. D. Fox, "Multipath sparse coding using hierarchical matching pursuit," in *CVPR*, 2013.

[27] J. van Gemert, C. Veenman, A. Smeulders, and J.-M. Geusebroek, "Visual word ambiguity," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2010.

[28] Y. Boureau, N. Le Roux, F. Bach, J. Ponce, and Y. LeCun, "Ask the locals: Multi-way local pooling for image recognition," in *ICCV*, 2011.

[29] H. Lee, C. Ekanadham, and A. Ng, "Sparse deep belief net model for visual area V2," in *NIPS*, 2008.

[30] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Supervised dictionary learning," in *NIPS*, 2008.

[31] Z. Jiang, Z. Lin, and L. S. Davis, "Learning a discriminative dictionary for sparse coding via label consistent K-SVD," in *CVPR*, 2011.

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

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

[34] Y. Boureau, F. Bach, Y. LeCun, and J. Ponce, "Learning mid-level features for recognition," in *CVPR*, 2010.

[35] J. Yang, K. Yu, and T. Huang, "Efficient highly over-complete sparse coding using a mixture model," in *ECCV*, 2010.

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

[37] H. Goh, N. Thome, and M. Cord, "Biasing restricted Boltzmann machines to manipulate latent selectivity and sparsity," in *NIPS Workshop*, 2010.

[38] H. Goh, L. Kusmierz, J.-H. Lim, N. Thome, and M. Cord, "Learning invariant color features with sparse topographic restricted Boltzmann machines," in *ICIP*, 2011.

[39] S. Lazebnik and M. Raginsky, "Supervised learning of quantizer codebooks by information loss minimization," *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 1294–1309, 2009.

[40] Y. Boureau, J. Ponce, and Y. LeCun, "A theoretical analysis of feature pooling in vision algorithms," in *ICML*, 2010.

[41] S. Avila, N. Thome, M. Cord, E. Valle, and A. Araujo, "Pooling in image representation: the visual codeword point of view," *Comput. Vis. Image. Und.*, 2012.

[42] F. Perronnin and C. Dance, "Fisher kernels on visual vocabularies for image categorization," in *CVPR*, 2007.

[43] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *CVPR*, 2010.

[44] X. Zhou, K. Yu, T. Zhang, and T. Huang, "Image classification using super-vector coding of local image descriptors," in *ECCV*, 2010.

[45] S. McCann and D. G. Lowe, "Spatially local coding for object recognition," in *ACCV*, 2012.

[46] V. N. Vapnik, *The nature of statistical learning theory*. New York, NY, USA: Springer-Verlag New York, Inc., 1995.

[47] K. Kavukcuoglu, M. Ranzato, R. Fergus, and Y. LeCun, "Learning invariant features through topographic filter maps," in *CVPR*, 2009.

[48] K. Yu, Y. Lin, and J. D. Lafferty, "Learning image representations from the pixel level via hierarchical sparse coding," in *CVPR*, 2011.

[49] M. Riesenhuber and T. Poggio, "Hierarchical models of object recognition in cortex," *Nat. Neurosci.*, vol. 2, pp. 1019–1025, 1999.

[50] S. Bileschi, M. Riesenhuber, T. Poggio, T. Serre, and L. Wolf, "Robust object recognition with cortex-like mechanisms," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, pp. 411–426, 2007.

[51] J. Mutch and D. Lowe, "Object class recognition and localization using sparse features with limited receptive fields," *Int. J. Comput. Vision*, vol. 80, pp. 45–47, 2008.

[52] C. Theriault, N. Thome, and M. Cord, "Extended coding and pooling in the hmax model," *IEEE Trans. Image Process.*, 2012.

[53] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Comput.*, vol. 14, no. 8, p. 1771–1800, 2002.

[54] G. Hinton, "A practical guide to training restricted Boltzmann machines," Dept. of Comp. Sci., University of Toronto, Tech. Rep. UTML TR 2010–003, 2010.

[55] K. Swersky, B. Chen, B. Marlin, and N. de Freitas, "A tutorial on stochastic approximation algorithms for training restricted boltzmann machines and deep belief nets," in *ITA Workshop*, 2010.

[56] D. Blackwell, "Conditional expectation and unbiased sequential estimation," *Ann. Stat.*, vol. 18, pp. 105–110, 1947.

[57] T. M. Mitchell, "The need for biases in learning generalizations," Departament of Computer Science, Rutgers University, Technical Report CBM-TR-117, 1980.

[58] B. Willmore and D. J. Tolhurst, "Characterizing the sparseness of neural codes," *Network: Comp. Neural*, vol. 12, no. 3, p. 255–270, 2001.
[59] H. B. Barlow, "Unsupervised learning," *Neural Comput.*, vol. 1, no. 3, pp. 295–311, 1989.
[60] E. T. Rolls and A. Treves, "The relative advantage of sparse versus distributed encoding for associative neuronal networks in the brain," *Network: Comp. Neural*, vol. 1, no. 4, pp. 407–421, 1990.
[61] P. Földiák, "Neural coding: non-local but explicit and conceptual," *Curr. Biol.*, vol. 19, no. 19, 2009.
[62] V. Nair and G. Hinton, "3D object recognition with deep belief nets," in *NIPS*, 2009.
[63] M. Welling, G. E. Hinton, and S. Osindero, "Learning sparse topographic representations with products of student-t distributions," in *NIPS*, 2003.
[64] A. Hyvärinen and P. O. Hoyer, "A two-layer sparse coding model learns simple and complex cell receptive fields and topography from natural images," *Vision Res.*, vol. 41, no. 18, pp. 2413–2423, 2001.
[65] J. Ngiam, P. W. Koh, Z. Chen, S. Bhaskar, and A. Ng, "Sparse filtering," in *NIPS*, 2011.
[66] H. Larochelle and Y. Bengio, "Classification using discriminative restricted Boltzmann machines," in *ICML*, 2008.
[67] H. Goh, N. Thome, M. Cord, and J.-H. Lim, "Top-down regularization of deep belief networks," in *Advances in Neural Information Processing Systems (NIPS)*, 2013.
[68] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories," in *CVPR Workshop*, 2004.
[69] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," California Institute of Technology, Tech. Rep. 7694, 2007. [Online]. Available: http://authors.library.caltech.edu/7694
[70] L. Liu, L. Wang, and X. Liu, "In defense of soft-assignment coding," in *ICCV*, 2011.
[71] G. L. Oliveira, E. R. Nascimento, A. W. Vieira, and M. F. M. Campos, "Sparse spatial coding: A novel approach for efficient and accurate object recognition," in *ICRA*, 2012.
[72] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus, "Deconvolutional networks," in *CVPR*, 2010.
[73] O. Duchenne, A. Joulin, and J. Ponce, "A graph-matching kernel for object categorization," in *ICCV*, 2011.
[74] J. Feng, B. Ni, Q. Tian, and S. Yan, "Geometric ℓ_p-norm feature pooling for image classification," in *CVPR*, 2011.
[75] O. Boiman, E. Shechtman, and M. Irani, "In defense of nearest-neighbor based image classification," in *CVPR*, 2008.
[76] T. Tuytelaars, M. Fritz, K. Saenko, and T. Darrell, "The NBNN kernel," in *ICCV*, 2011.
[77] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng, "Self-taught learning: Transfer learning from unlabeled data," in *ICML*, 2007.

Hanlin Goh received the B.Eng. degree in computer engineering and the M.Sc. degree in bioinformatics from the Nanyang Technological University, Singapore, in 2007 and 2009 respectively, and a Ph.D. degree (with highest honors) in computer science from the Université Pierre et Marie Curie – Sorbonne Universités, Paris, France, in 2013.

He is currently a Research Scientist with the Visual Computing Department at the Institute for Infocomm Research, A*STAR, Singapore, and a member of the Image and Pervasive Access Laboratory, CNRS UMI 2955, a French-Singaporean joint laboratory. His research motivation is to design machine learning algorithms that lead to scalable solutions for broad competence computer vision.

Nicolas Thome (M'10) received the diplôme d'Ingénieur from the École Nationale Supérieure de Physique de Strasbourg, France, the DEA (MSc) degree from the University of Grenoble, France, in 2004 and, in 2007, the PhD degree in computer science from the University of Lyon, France. In 2008, he was a postdoctoral associate at INRETS in Villeneuve d'Ascq, France. Since 2008 is an assistant professor at Université Pierre et Marie Curie (UPMC) and Laboratoire d'Informatique de Paris 6 (LIP6). His research interests are in the area of Computer Vision and Machine Learning, particularly in the design and learning of complex image representations and similarities, with applications to image and video understanding.

Matthieu Cord (M'09) received the Ph.D. degree in computer science from the UCP, France, before working in the ESAT lab at KUL University, Belgium, and in the ETIS lab, France as Assistant Professor. He joined the Computer Science department LIP6, at UPMC Sorbonne Universitiés, Paris, in 2006 as full Professor. In 2009, he was nominated at the IUF (French Research Institute) for a 5 years delegation position.

His research interests include Computer Vision, Image Processing, and Pattern Recognition. He developed several systems for content-based image and video retrieval, focusing on interactive learning-based approaches. He is also interested in Machine Learning for Multimedia processing, Digital preservation, and Web archiving.

Prof. Cord has published a hundred scientific publications and participated in several international projects (European FP6 and FP7, Singapore, Brazil) on these topics. He is a member of the IEEE.

Joo-Hwee Lim (M'07) received his B.Sc. (First Class Honours) and M.Sc. (by research) degrees in Computer Science from the National University of Singapore and his Ph.D. degree in Computer Science & Engineering from the University of New South Wales.

He is currently the Head of the Visual Computing Department at the Institute for Infocomm Research (I2R), A*STAR, Singapore, and an Adjunct Associate Professor at the School of Computer Engineering, Nanyang Technological University, Singapore. He is the co-Director of IPAL (Image & Pervasive Access Laboratory), a French-Singapore Joint Lab (CNRS UMI 2955, Jan 2007 to Jan 2015). He is bestowed the title of 'Chevalier dans lordre des Palmes Academiques' by the French Government in 2008 and the National Day Commendation Medal by the Singapore Government in 2010. He has published more than 190 international refereed journal and conference papers and co-authored 18 patents (awarded and pending) in his research areas of computer vision, cognitive vision, pattern recognition, and medical image analysis.