Unsupervised and Supervised Visual Codes with Restricted Boltzmann Machines

Hanlin Goh[1,2,3], Nicolas Thome[1], Matthieu Cord[1], and Joo-Hwee Lim[1,2,3]

[1] Laboratoire d'Informatique de Paris 6, UMPC - Sorbonne Universités, France
[2] Institute for Infocomm Research, A*STAR, Singapore
[3] Image and Pervasive Access Laboratory, CNRS UMI 2955, France and Singapore
{hanlin.goh,nicolas.thome,matthieu.cord}@lip6.fr, joohwee@i2r.a-star.edu.sg

Abstract. Recently, the coding of local features (e.g. SIFT) for image categorization tasks has been extensively studied. Incorporated within the Bag of Words (BoW) framework, these techniques optimize the projection of local features into the visual codebook, leading to state-of-the-art performances in many benchmark datasets. In this work, we propose a novel visual codebook learning approach using the restricted Boltzmann machine (RBM) as our generative model. Our contribution is three-fold. Firstly, we steer the unsupervised RBM learning using a regularization scheme, which decomposes into a combined prior for the sparsity of each feature's representation as well as the selectivity for each codeword. The codewords are then fine-tuned to be discriminative through the supervised learning from top-down labels. Secondly, we evaluate the proposed method with the Caltech-101 and 15-Scenes datasets, either matching or outperforming state-of-the-art results. The codebooks are compact and inference is fast. Finally, we introduce an original method to visualize the codebooks and decipher what each visual codeword encodes.

1 Introduction

Image classification is one of the most challenging problems in computer vision, since it implies predicting complex semantic categories, like scenes or objects, from raw pixels. Although the ultimate solution to bridge this semantic gap remains unclear, two major developments have emerged in the last decade towards this goal. The first is the design of powerful low-level local descriptors, such as SIFT [1]. The second is the notion of mid-level representations inspired from the text retrieval community, based on the Bag of Words (BoW) model [2]. In the BoW model, converting the set of local descriptors into the final image vectorial representation is performed by a succession of two steps: coding and pooling.

The coding step executes a hard quantization of features, such as nearest neighbor and sum pooling. In computer vision, however, the notion of 'word' is not as well defined as in text retrieval, resulting in ambiguous local descriptor to visual word assignment. Improving the coding with the goal of attenuating the quantization loss has garnered recent attention. An alternative known as soft assignment smoothly distributes features to the codebook elements [3, 4]. A variety of machine learning techniques that rely on supervision [5–9] or otherwise, such as sparse coding [10–12] and restricted Boltzmann machines (RBM) [13, 14], have been adapted to tackle the codebook learning problem.

Another flaw of the BoW model is the dissipation of spatial information resulting from pooling over the whole image. To address this, the spatial pyramid scheme [15] computes histograms in fixed multi-resolution spatial grids over the image. Different pooling strategies have also been studied to improve the final image representation used for classification. Max-pooling appears to be a promising alternative to sum-pooling, especially when linear classifiers are used [9, 16]. Therefore, the combination of spatial pyramids and max-pooling is often regarded as the desired strategy leading to state-of-the-art performances. In this work, we focus on the coding step prior to the construction of spatial pyramids.

2 Related Work and Contributions

Let us consider a given local descriptor $\mathbf{x} \in \mathbb{R}^I$ and a matrix $\mathbf{W} \in \mathbb{R}^{I \times J}$. Sparse coding is a decoder architecture that 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 prior that promotes sparse solutions:

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

where λ is a constant and the ℓ_1 norm is often used to approximate of the ℓ_0 norm, leading to a convex problem. Generating the coding for each descriptor requires solving Eq. 1. The inference problem quickly becomes a computational challenge, especially when the number of descriptors or the codebook size J increases.

Several approaches for limiting the computational cost of sparse coding have been proposed. Yang et al. [17] used a mixture model to represent low-level descriptors, so that the optimization is limited to within a mixture component. Bourreau et al. [11] suggested using two dictionaries: a small one for sparse coding and the other to incorporate locality constraints during pooling. With the best results obtained using a vector of greater than 1 million dimensions, this strategy sacrifices representational compactness to gain encoding speed. Kavukcuoglu et al. [12], on the other hand, learned an encoder-decoder network with an added encoding term to Eq. 1. The burden of the heavy minimization step during inference is transferred to the encoder during learning. However, their model is learned from the pixel-level, making it challenging to achieve competitive performances in benchmarking datasets, such as Caltech 101 [18].

In this paper, our framework is based on restricted Boltzmann machines (RBM), which are encoder-decoder networks. RBMs are faster than sparse coding [10, 11] during inference, as there are no optimization steps involved. Furthermore, RBMs naturally encode locality, transferring the similarity between descriptors from feature space to coding space [19]. RBMs have also been stacked to form hierarchical representations from pixels [20], with selectivity [13] as the prior to train each layer. Sohn et al. [14] extended the work by learning Gaussian

RBMs from SIFT rather than pixels, but the overall architecture remains relatively heavy. A separate line of work focuses on biasing RBMs with task-specific properties, such as topographical representations that model transformation invariance [21]. However, existing work only explored image patches as inputs and fall short of attempting image classification within the BoW model.

Sparse coding and RBMs build their codebooks purely from the bottom-up via unsupervised learning without regard of the task. Another class of methods exploit labeled data to increase feature discrimination. Typically, the learning algorithm incorporates an additional supervised term in the optimization, such as mutual information loss [22]. The methods may be classified with respect to the level where discrimination is incorporated. Some methods directly use a discriminative criterion for each local descriptor [5, 6]. Since an image label usually does not propagate to every local portion of the image, these methods are more likely to suffer from noise. Other methods [7–9] use a discriminative term that associates labels to a spatially-pooled global image statistic. This information is "un-pooled" to optimize the codebook parameters and are generally complex and slow. For example, in Yang et al. [8], backpropagation can only be executed stochastically due to pooling, resulting in long training times.

Contributions. In this paper, we propose to learn visual codes from SIFT descriptors using RBMs trained over two phases – a unsupervised learning phase and a supervised fine-tuning phase. The overall architecture follows the standard BoW pipeline. Our contributions are the following:

- During unsupervised learning of local descriptors, we employ a RBM regularization method that jointly enforces sparse representations and selective codewords. For the supervised phase, the codewords are adapted to be discriminative with respect to a local classifier that is concurrently learned.
- Our method leads to very competitive results, either reaching or surpassing the state-of-the-art on both the Caltech-101 and the 15-Scenes datasets. Our image representation is more compact than those of recent methods [11, 14]. Being an encoder-decoder network, the RBM only requires a simple feed-forward projection step during inference. This is extremely fast – we observe a boost of at least 200 times relative to decoder-only methods [10, 23, 9, 11].
- Finally, we introduce a new method to visualize the learned codewords for local gradient-based features, such as SIFT.

3 Visual Codebook Learning

Fig. 1 shows our BoW framework using restricted Boltzmann machines (RBM) to learn visual codes and perform feature coding during inference. During training, the restricted Boltzmann machine (RBM) is first used to perform unsupervised learning of bottom-up local features (Section 3.1). The learning is concurrently regularized to guide the representation to be sparse and codewords to be selective (Section 3.2). The second phase uses top-down category labels to adjust the

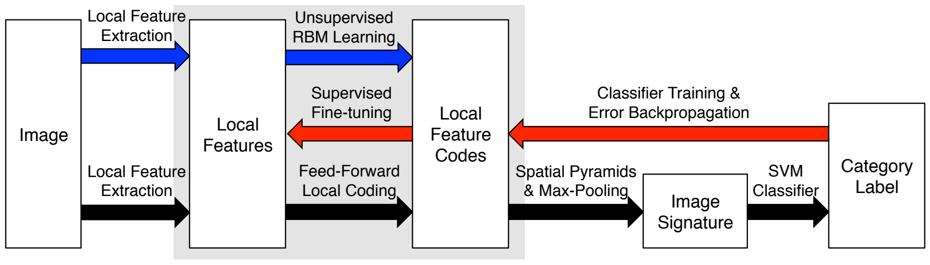


Fig. 1. Our unsupervised and supervised RBM-based BoW architecture consists of five layers of representations. The arrows indicate the operations performed during various phases: 1) unsupervised learning (blue), 2) supervised fune-tuning (red), and 3) SVM classifier learning and classification (black). This paper focuses on the mapping between the local feature layer and the local coding layer (shaded)

learned codebook. We concurrently learn a local classifier, while backpropagating the residual error to fine-tune codebook (Section 3.3). After training, the RBM is used as a feed-forward encoder to directly infer the coding of each local feature. Spatial pyramids [15] and max-pooling are then used to introduce spatial coding and form the global image signature. Finally, a support vector machine (SVM) is trained to classify signatures into their respective categories.

3.1 Unsupervised Learning with Restricted Boltzmann Machines

An RBM is a fully connected bipartite graph with one input feature layer $\mathbf{x}$ and one latent coding layer $\mathbf{z}$. The feature layer contains I dimensions corresponding to the size of the input vector (e.g. 128 for SIFT). The coding layer has J visual codewords used to encode the feature layer.

The layers are connected via undirected weights $\mathbf{W} \in \mathbb{R}^{I \times J}$, our visual codebook. Additionally, each unit also receives input from a bias – c_i or b_j for the feature layer and coding layer respectively. For a binary RBM, the activation probabilities of units in one layer can be computed by sampling from the opposite layer with a sigmoid activation function as follows:

$$P\left(z_j \mid \mathbf{x}\right) = sigmoid\left(b_j + \sum_{i=1}^{I} w_{ij} x_i\right), \tag{2}$$

$$P\left(x_i \mid \mathbf{z}\right) = sigmoid\left(c_i + \sum_{j=1}^{J} w_{ij} z_j\right), \tag{3}$$

where x_i and z_j refer to the state of units in the feature and coding layers respectively. Given a visual codebook $\mathbf{W}$ and biases $\mathbf{b}$, feature $\mathbf{x}$ can be encoded to its visual code $\mathbf{z}$. Likewise, if we have the latent representation $\mathbf{z}$ and parameters $\mathbf{W}$ and $\mathbf{c}$, the feature $\mathbf{x}$ can be reconstructed.

The negative log likelihood of states in the RBM can be formulated as an energy function to be minimized:

$$E(\mathbf{x},\mathbf{z}) = -\log P(\mathbf{x},\mathbf{z}) = -\sum_{i=1}^{I}\sum_{j=1}^{J} x_i w_{ij} z_j - \sum_{i=1}^{I} c_i x_i - \sum_{j=1}^{J} b_j z_j. \tag{4}$$

The parameters $\mathbf{W}$, $\mathbf{b}$ and $\mathbf{c}$, are trained with the objective of minimizing the negative likelihood of the data distribution. The optimization is approximated using the contrastive divergence (CD) algorithm [24], whereby the joint states randomly sampled from the data distribution are contrasted against the "fantasy" states that the network believes more than real training data. The estimator is obtained by performing a finite (usually small) number of alternating Gibbs sampling iterations between the two layers. To produce an estimator with lower variance [25], we use the activation probabilities of x_i and z_j instead of their binary states when updating the parameters (also see [26]).

3.2 Sparse and Selective Regularization

When learning mid-level representations, we want the encoded feature to exhibit certain characteristics that will ultimately be useful for the image categorization task. This motivates us to regularize the learning of the RBM, by appending a regularization term $h(\mathbf{z})$ weighted by λ to the RBM optimization problem:

$$\underset{\{\mathbf{W},\mathbf{c},\mathbf{b}\}}{\arg\min} \quad -\sum_{k=1}^{K} \log \sum_{\mathbf{z}} \Pr(\mathbf{x}_k, \mathbf{z}_k) + \lambda h(\mathbf{z}). \tag{5}$$

We explore the notion of sparsity and selectivity in visual coding. Sparsity is a metric of the entire coding vector in response to one input feature; selectivity measures the response of a single visual codeword across a set of input features.

A coarse method to quantify selectivity or sparsity is take the mean response of codewords in either dimension (lifetime for selectivity and population for sparsity). This metric has been exploited to encourage selectivity in RBM learning by penalizing codewords that have high average activations [13]:

$$h(\mathbf{z}) = \sum_{j=1}^{J} \left\| \hat{p} - \frac{1}{K}\sum_{k=1}^{K} z_{jk} \right\|^2, \tag{6}$$

where $\hat{p}$ is the desired (usually low) mean activation probability of each codeword across K examples. A conceptually similar approach of penalizing activation averages across examples have also been suggested [27].

However, simplicity has its drawbacks. This method penalizes a codeword uniformly across the batch of examples, regardless of its supposed selectivity to individual examples. Moreover, a low mean activation could be achieved when collectively the activations are not selective, for example, when the activations are individually equal to the mean $\hat{p}$. In this case, the regularization signal may be erroneous. Moreover, the formulation supports only selectivity but not sparsity.

During feature coding, each visual codeword should respond to only a small subset of input feature descriptors. Similarly, we want only a small subset of codewords to respond to each feature. When these two requirements are combined, we promote diversity among the codewords and differentiation between the representation of different features. This need for a feature-wise and codeword-wise control of regularization for both sparsity and selectivity is the motivation behind the precise regularization of the RBM.

The aim is to influence the activation of individual codewords j for in response to each input sample k. The desired coding of feature k by codeword z_j can be denoted by $p_{jk} \in [0, 1]$, which can be organized into a target matrix $\mathbf{P} \in \mathbb{R}^{J \times K}$. Each row $\mathbf{p}_j$ represents the desired activation vector of codeword z_j with respect to the set of K input features. Each column $\mathbf{p}_k$ denotes the population coded representation of feature k. To regularize the RBM, $h(\mathbf{z})$ can be defined using the cross-entropy loss, summed over J codewords and K features:

$$h(\mathbf{z}) = -\sum_{j=1}^{J}\sum_{k=1}^{K} p_{jk} \log z_{jk} + (1 - p_{jk}) \log (1 - z_{jk}) \,. \tag{7}$$

Both selectivity and sparsity are induced during learning by crafting appropriate targets $\mathbf{P}$. We regularize the responses of each codeword to match a selective activation distribution and each coding vector to induce sparsity. From a given array of observed data-sampled activations $\mathbf{z} \in \mathbb{R}^N$ (Fig. 2(a)), we first transform the activations to fit a uniform distribution (Fig. 2(d)) and subsequently obtain the $\mathbf{P}$ (Fig. 2(e)) by mapping to our target distribution (Fig. 2(f)). The two-step transformation is formulated as follows:

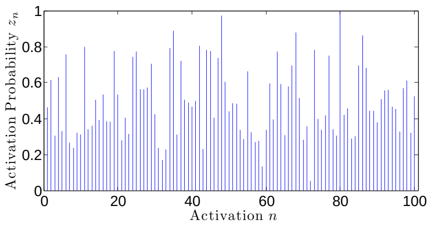
$$p_n = (rank(z_n, \mathbf{z}))^{\frac{1}{\mu}-1}, \tag{8}$$

where $rank(z_n, \mathbf{z})$ assigns 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 assigned as 1. The target mean $\mu \in (0, 1)$ positively skews the distribution when $\mu < 0.5$. With distribution based targets, some activations will be pulled up while most will be encouraged to be lower. This is as opposed to penalizing activations equally using a mean-based regularization [13, 27].

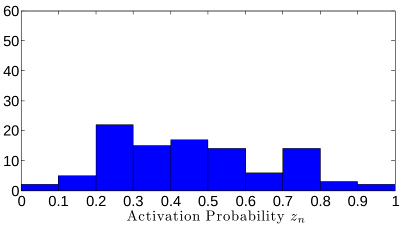
To obtain a jointly sparse and selective $\mathbf{P}$, the activations are mapped first by rows, then by columns [28, 29]. Due to the dependance of $\mathbf{P}$ on $\mathbf{Z}$, we alternate between computing the targets and updating the parameters.

3.3 Supervised Fine-Tuning

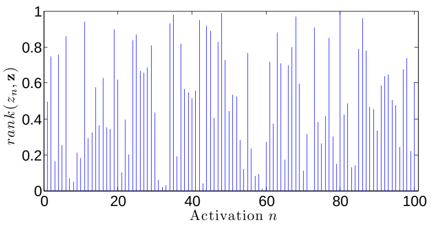
After unsupervised learning, we fine-tune the codebooks through supervised learning. We associate each local feature to the image label and modify the codewords to be discriminative with respect to this association. Although we do not pose the exact problem of whole image categorization, we want to improve local feature discrimination to boost the performances for classifying whole images. We present the results with and without supervision in the next section.



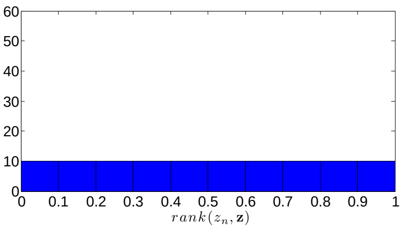
(a) Observed data-sampled activations



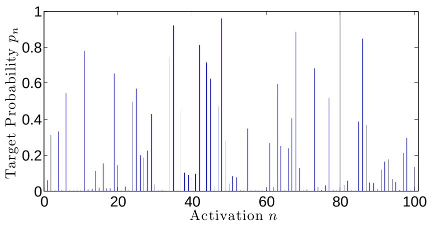
(b) Original distribution



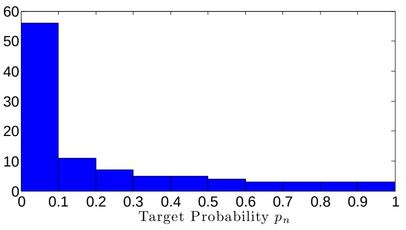
(c) After histogram equalization (ranked)



(d) Uniform distribution



(e) Sparse or selective target activations



(f) Positively skewed distribution

Fig. 2. On the left, the process of mapping a hypothetical set of 100 activations from (a) data-sampled to (c) ranked and finally to (e) sparse or selective target coding. The distributions of the respective activations are shown on the right. A few activations are encouraged to be high, but most have significantly lower target responses

The learned RBM is essentially a two-layer neural network, with initialized parameters that can be fine-tuned via supervised learning. The set of parameters $(\mathbf{W}, \mathbf{b})$ map from the feature layer $\mathbf{x}$ to the mid-level representation $\mathbf{z}$. For an image categorization task with L classes, a second set of randomly initialized weights $\mathbf{M} \in \mathbb{R}^{J \times L}$ and biases $\mathbf{d} \in \mathbb{R}^{L}$ transform $\mathbf{z}$ to the one-hot coded category labels $\mathbf{y} \in \mathbb{R}^{L}$ via feed-forward activations:

$$y_l = sigmoid\Big(d_l + \sum_{j=1}^{J} m_{jl} z_j\Big). \tag{9}$$

This connectionist architecture makes it simple to introduce supervision with the error backpropagation algorithm, which is designed to optimize multi-layered feed-forward networks. The error E between the hypothesized class $\mathbf{y}$ and the ground truth $\hat{\mathbf{y}}$ is defined as

$$E(\mathbf{W}, \mathbf{b}, \mathbf{M}, \mathbf{d} \mid \mathbf{x}) = \frac{1}{2} \left\| \hat{\mathbf{y}} - \mathbf{y} \right\|^2 . \tag{10}$$

A gradient descent with respect to the parameters across the two layers can be derived. In essence, we are trying to learn a simple one-layer classifier of local representations $\mathbf{z}$, while fine-tuning parameters $(\mathbf{W}, \mathbf{b})$ of the previous layer based-on residual error backpropagated from the top-down. After fine-tuning, the second-layer parameters $\mathbf{M}$ and $\mathbf{d}$ are discarded since they are no longer needed for inference. The final codebook is the result of the two-phase unsupervised and supervised learning process.

4 Experiment Results and Discussion

4.1 Experimental Setup

For this work, we focus on image categorization on the Caltech-101 dataset [18] and the 15-Scenes [15] dataset. The mean class-wise accuracy was used as the performance metric. The results were compiled over 10 trials.

The images were first resized to a maximum of 300×300 pixels.We experimented with two types of low-level local features: 1) SIFT [1] and 2) macrofeature (MF) [9]. SIFT descriptors scaled at 16×16 pixels were densely sampled from each image at 8 pixel intervals. Macrofeatures were pooled from 2×2 neighborhoods of SIFTs extracted at 4 pixel intervals.

A set of 200,000 randomly selected features were used to train the sparse and selective RBM with 1024 codewords. The trained codebook was used to encode the local features (SIFT or macrofeature). The macrofeatures codebooks were also fine-tuned through supervision. For each trial, a randomly selected set of the training images (either 15 or 30 per category for Caltech-101, and 100 for 15-Scenes) were used for supervised fine-tuning as well as SVM classifier training. The remaining images were used for testing.

From the visual codes, a three-level spatial pyramid was employed using standard pooling grids of 4×4, 2×2 and 1×1 to construct the final image signature. Finally, we trained a linear SVM to perform multi-class classification of the test images.

4.2 Image Categorization Accuracy

Our best image categorization results obtained on the Caltech-101 dataset were $71.1 \pm 1.3\%$ and $78.9 \pm 1.1\%$, using 15 and 30 training images. This was achieved with 1024 codewords trained on macrofeatures with supervised fine-tuning. For the 15-Scenes dataset, the macrofeatures codebook regularized with sparse and selective performs extremely well, obtaining a result of $85.7 \pm 0.7\%$, which was further boosted to $86.0 \pm 0.5\%$ after supervised fine-tuning.

The macrofeature descriptor consistently outperforms SIFT descriptors, by about 3% on the Caltech-101 dataset and 1.5% for the 15-Scenes dataset. This improvement in performance validates the results reported by Boureau et al. [9].

Comparison with Other Methods. In Table 1, we compare our results with other feature coding strategies following the same BoW pipeline that use a single feature type for image categorization. For all three setups, we achieved the state-of-the-art results after supervised fine-tuning is performed. Even without supervision, our architecture managed to maintain its competitiveness. It also appears that RBM-based methods are slowly gaining a competitive edge over its assignment coding and sparse coding counterparts.

Table 2 shows the results of non-coding-based methods also using a single feature type, but focusing on aspects of image categorization other than coding. In particular, two recent methods by Duchenne et al. [30] and Feng et al. [31] reported impressive performances of $80.3 \pm 1.2\%$ and 82.6% respectively, on the Caltech-101 dataset using 30 training examples. Duchenne et al. [30] used graph matching to encode the spatial information of sparse codes [9], while Feng et al. [31] build upon LLC sparse codes [23] to perform pooling in a discriminative manner. These two methods address the categorization problem in a complete different direction as we do. Our methods are complementarily, with the possibility to make further performance improvements if combined.

Table 1. Performance comparison with BoW coding methods

Method	Codebook Size	Caltech-101 15 tr.	Caltech-101 30 tr.	15-Scenes 100 tr.
Non-Learned Assignment Coding				
Hard assignment [15]	200	56.4	64.6 ± 0.8	81.1 ± 0.3
Kernel codebooks [3]	200	-	64.1 ± 1.5	76.7 ± 0.4
Soft assignment [4]	1000	-	74.2 ± 0.8	82.7 ± 0.4
Sparse Coding				
ScSPM [10]	1024	67.0 ± 0.5	73.2 ± 0.5	80.3 ± 0.9
LLC [23]	2048	65.4	73.4	-
Sparse codes & max-pooling [9]	1024	-	75.7 ± 1.1	84.3 ± 0.5
Multi-way local pooling [11]	1024×65	-	77.3 ± 0.6	83.1 ± 0.7
Restricted Boltzmann Machine				
CDBN [20]	200	57.7 ± 1.5	65.4 ± 0.5	-
Sparse RBM [14]	4096	68.6	74.9	-
CRBM [14]	4096	71.3	77.8	-
Supervised Learning				
Discriminative dictionary [9]	2048	-	-	85.6 ± 0.2
LC-KSVD [6]	1024	67.7	73.6	-
Our Method				
Unsupervised SS-RBM (SIFT)	1024	66.8 ± 1.6	75.1 ± 1.2	84.1 ± 0.8
Unsupervised SS-RBM (MF)	1024	70.2 ± 1.9	78.0 ± 1.4	85.7 ± 0.7
Supervised SS-RBM (MF)	1024	71.1 ± 1.3	78.9 ± 1.1	86.0 ± 0.5

Table 2. Results of other non-coding-based approaches, such as kernel methods

Method	Caltech-101		15-Scenes
	15 tr.	30 tr.	100 tr.
NBNN[32]	65.0 ± 1.1	70.4	75 ± 3
NBNN kernel [33]	69.2 ± 0.9	75.2 ± 1.2	85 ± 4
Hierarchical Gaussianization [34]	-	-	85.2
Graph-matching kernel [30]	75.3 ± 0.7	80.3 ± 1.2	82.1 ± 1.1
GLP [31]	70.3	82.6	83.2

Impact of Supervised Fine-Tuning. Supervised fine-tuning was performed on codebooks trained on macrofeatures. As reported in Table 1, when supervision is used, there is a slight improvement of classification accuracy in all datasets. An analysis of individual trials with 30 training examples on Caltech-101 (Fig. 3) revealed that every experimental trial is improved through fine-tuning. For example, the average improvement per trial is $0.9 \pm 0.6\%$ – a statistically significant gain. Our results are marginally better than those of the discriminative dictionary [9], which optimizes for global labeling via a more complex process as compared to our local labeling method. We also outperform LC-KSVD [6], mainly due to superior unsupervised learning and local features.

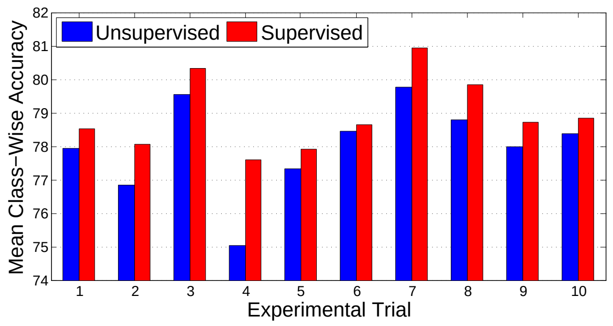


Fig. 3. Results of 10 trials on Caltech-101 (30 examples) with unsupervised learning only (blue) and adding supervision (red) shows every trial improving after fine-tuning

4.3 Computational Resources

Representation Compactness. For a fair comparison with other methods, we show in Fig. 4 the results of a SIFT codebook trained on Caltech-101 (30 examples) only using unsupervised RBM learning regularized with sparsity and selectivity. Our method remains very competitive even as the codebook size is reduced by four times from 4096 to 1024. With only 128 codewords, we are still able to get accuracies greater than 60%. We observed that a larger codebook has more capacity to capture the diversity in the features, but it is also more likely to exhibit codeword redundancy. From our experiments, 1024 codewords appear to give a good balance between diversity and conciseness.

The results using soft assignment [4] is $74.2 \pm 0.8\%$. With the same codebook size, we manage to achieve a higher accuracy of $78.0 \pm 1.4\%$ purely with unsupervised learning. Our result matches that of the CRBM [14] (77.8%), but we used only a quarter of the codewords. Our performance also mirrors that of Boureau et al. [11] of $77.3 \pm 0.6\%$, although due to their pooling strategy, the final image signature is 65 times larger than ours.

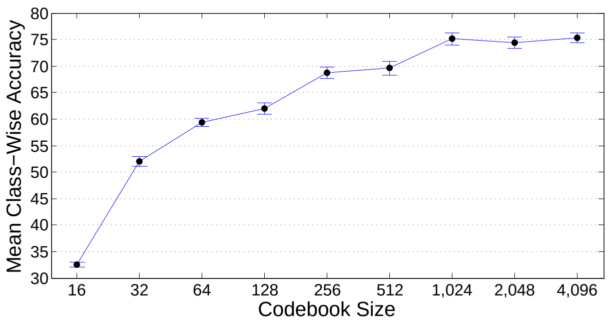


Fig. 4. Classification results of an unsupervised SIFT codebook Caltech-101 (30 training examples) with varying number of codewords. The performance degrades but remains competitive even when the number of codewords is dramatically reduced

Inference Speed. Feature coding is fast during inference because we exploit the encoder nature of the RBM. The inference time for the architecture is the same, no matter supervised or unsupervised, because we merely modify existing parameters during supervised learning. For each descriptor, only a feed-forward computation needs to be performed to obtain its coding. Batches of descriptors can be computed concurrently with suitable implementations. The advantage of inference speed is significant when compared against sparse coding methods, which have to run the sparse optimization during inference. Experimentally, we observed a speedup of at least 200 times as compared to the ScSPM method [10].

4.4 Visualization of Codewords

Neural networks are often considered to be black boxes, with low interpretability of what is learned. However, sparse RBMs that model image pixels are able to visualize the layer of units as a collection of bases that resemble Gabor filters [13].

Here, we want to visualize the result of unsupervised learning of local gradient features, such as SIFT. Each codeword can be extracted as a 'filter' over the input SIFT feature space. Since each descriptor is split into a 4×4 neighborhood grid, from which local gradients are computed, we partition each filter in the same manner. We assign to each partition its dominant orientation with a strength proportional to its response.

The resulting visualization is a filter map as shown in Figure 5. In many codewords, opposing gradients are paired (i.e. red-cyan, orange-blue, yellow-purple, green-magenta. Each pairing also has a consistent direction, for example red-cyan pairings only occur left and right of each other and orange-blue pairing occurs at around 135 and 315 degrees. It is interesting that such coherent structures can be automatically discovered by the regularized RBM. The codewords appear to encode generic image structure. When few codewords are used, they tend to be very diverse, resulting in concise codebooks. As the size of codebook increases, more redundancy is built into the codebook, as it gets increasingly difficult to discover features that are very different from one another. This may be the reason why the performances of smaller codebooks are relatively high, while the results of larger codebooks tend to stagnate.

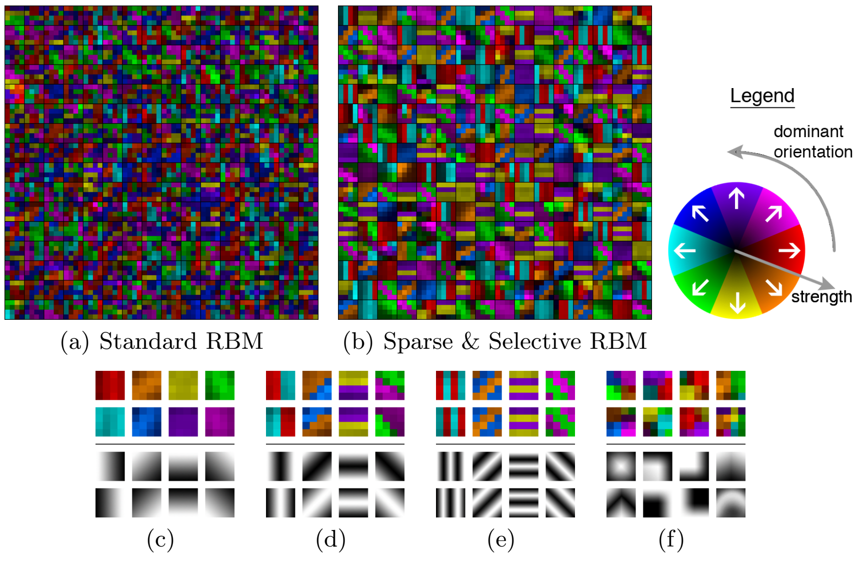


Fig. 5. Visualization of some visual codewords learned from SIFT using (a) standard RBMs and (b) sparse and selective RBMs. Each square represents a visual codeword projected back into the feature space. Each codeword is further partitioned into 4×4 neighborhoods, each displaying its dominant orientation as one of eight colors, while the response strength of the partition is indicated by its color intensity (see legend). Our codewords capture spatially coherent structure, encoding a variety of (c) smooth gradients, (d) lines and edges, (e) textured gratings and (f) other complex features, such as corners, bends and center-surround features

5 Conclusion

In this paper, we used the RBMs to learn visual codebooks from SIFT features, within the BoW framework. The visual codebooks are trained in two learning phases - unsupervised and supervised. The unsupervised learning is designed to maximize the likelihood of data, while being regularized to be jointly sparse and selective. The supervised phase is executed by error backpropagation, which concurrently learns the mapping from the coding layer to the labels, while fine-tuning the codebook parameters.

The image categorization results achieved on benchmarking datasets (Caltech-101 and 15-Scene) are extremely competitive with other state-of-the-art methods. It performs surprisingly well even with small codebooks. Feature coding during inference is fast because the RBM acts as a simple feed-forward encoder. Through codebook visualization, we discover that the codebooks are concise and capture the underlying structure of gradients in the image.

Although the supervised optimization for associating local descriptors to labels is simple, it deviates from the actual problem of global image categorization. Future work to improve the framework should attempt to alleviate the limitations by exploring computationally efficient and lossless pooling methods.

References

1. Lowe, D.: Distinctive image features from scale-invariant keypoints. IJCV **60** (2004) 91–110
2. Sivic, J., Zisserman, A.: Video Google: A text retrieval approach to object matching in videos. In: ICCV. (2003)
3. van Gemert, J., Veenman, C., Smeulders, A., Geusebroek, J.M.: Visual word ambiguity. PAMI (2010)
4. Liu, L., Wang, L., Liu, X.: In defense of soft-assignment coding. In: ICCV. (2011)
5. Mairal, J., Bach, F., Ponce, J., Sapiro, G., Zisserman, A.: Supervised dictionary learning. In: NIPS. (2008)
6. Jiang, Z., Lin, Z., Davis, L.S.: Learning a discriminative dictionary for sparse coding via label consistent K-SVD. In: CVPR. (2011)
7. Yang, L., Jin, R., Sukthankar, R., Jurie, F.: Unifying discriminative visual codebook generation with classifier training for object category recognition. In: CVPR. (2008)
8. Yang, J., Yu, K., Huang, T.: Supervised translation-invariant sparse coding. In: CVPR. (2010)
9. Boureau, Y., Bach, F., LeCun, Y., Ponce, J.: Learning mid-level features for recognition. In: CVPR. (2010)
10. Yang, J., Yu, K., Gong, Y., Huang, T.: Linear spatial pyramid matching using sparse coding for image classification. In: CVPR. (2009)
11. Boureau, Y., Le Roux, N., Bach, F., Ponce, J., LeCun, Y.: Ask the locals: Multi-way local pooling for image recognition. In: ICCV. (2011)
12. Kavukcuoglu, K., Sermanet, P., Boureau, Y., Gregor, K., Mathieu, M., LeCun, Y.: Learning convolutional feature hierachies for visual recognition. In: NIPS. (2010)
13. Lee, H., Ekanadham, C., Ng, A.: Sparse deep belief net model for visual area V2. In: NIPS. (2008)

14. Sohn, K., Jung, D.Y., Lee, H., Hero III, A.: Efficient learning of sparse, distributed, convolutional feature representations for object recognition. In: ICCV. (2011)
15. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: CVPR. (2006)
16. Boureau, Y., Ponce, J., LeCun, Y.: A theoretical analysis of feature pooling in vision algorithms. In: ICML. (2010)
17. Yang, J., Yu, K., Huang, T.: Efficient highly over-complete sparse coding using a mixture model. In: ECCV. (2010)
18. Fei-Fei, L., Fergus, R., Perona, P.: Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In: CVPR Workshop. (2004)
19. Salakhutdinov, R., Hinton, G.: Semantic hashing. International Journal of Approximate Reasoning **50** (2009) 969–978
20. Lee, H., Grosse, R., Ranganath, R., Ng, A.Y.: Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In: ICML. (2009)
21. Goh, H., Kusmierz, L., Lim, J.H., Thome, N., Cord, M.: Learning invariant color features with sparse topographic restricted Boltzmann machines. In: ICIP. (2011)
22. Lazebnik, S., Raginsky, M.: Supervised learning of quantizer codebooks by information loss minimization. PAMI (2009) 1294–1309
23. Wang, J., Yang, J., Yu, K., Lv, F., Huang, T., Gong, Y.: Locality-constrained linear coding for image classification. In: CVPR. (2010)
24. Hinton, G.E.: Training products of experts by minimizing contrastive divergence. Neural Computation **14** (2002) 1771–1800
25. Swersky, K., Chen, B., Marlin, B., de Freitas, N.: A tutorial on stochastic approximation algorithms for training restricted boltzmann machines and deep belief nets. In: ITA Workshop. (2010)
26. Hinton, G.: A practical guide to training restricted boltzmann machines. Technical Report UTML TR 2010–003, Dept. of Comp. Sci., University of Toronto (2010)
27. Nair, V., Hinton, G.: 3D object recognition with deep belief nets. In: NIPS. (2009)
28. Goh, H., Thome, N., Cord, M.: Biasing restricted Boltzmann machines to manipulate latent selectivity and sparsity. In: NIPS Workshop. (2010)
29. Ngiam, J., Koh, P.W., Chen, Z., Bhaskar, S., Ng, A.: Sparse filtering. In: NIPS. (2011)
30. Duchenne, O., Joulin, A., Ponce, J.: A graph-matching kernel for object categorization. In: ICCV. (2011)
31. Feng, J., Ni, B., Tian, Q., Yan, S.: Geometric ℓ_p-norm feature pooling for image classification. In: CVPR. (2011)
32. Boiman, O., Shechtman, E., Irani, M.: In defense of nearest-neighbor based image classification. In: CVPR. (2008)
33. Tuytelaars, T., Fritz, M., Saenko, K., Darrell, T.: The NBNN kernel. In: ICCV. (2011)
34. Zhou, X., Cui, N., Li, Z., Liang, F., Huang, T.: Hierachical Gaussianization for image classification. In: ICCV. (2009)