Modeling Pixel Means and Covariances Using Factorized Third-Order Boltzmann Machines

Marc'Aurelio Ranzato Geoffrey E. Hinton
Department of Computer Science - University of Toronto
10 King's College Road, Toronto, Canada
{ranzato,hinton}@cs.toronto.edu

Abstract

Learning a generative model of natural images is a useful way of extracting features that capture interesting regularities. Previous work on learning such models has focused on methods in which the latent features are used to determine the mean and variance of each pixel independently, or on methods in which the hidden units determine the covariance matrix of a zero-mean Gaussian distribution. In this work, we propose a probabilistic model that combines these two approaches into a single framework. We represent each image using one set of binary latent features that model the image-specific covariance and a separate set that model the mean. We show that this approach provides a probabilistic framework for the widely used simple-cell complex-cell architecture, it produces very realistic samples of natural images and it extracts features that yield state-of-the-art recognition accuracy on the challenging CIFAR 10 dataset.

1. Introduction

Many computer vision algorithms are based on probabilistic models of natural images that are used to provide sensible priors for tasks such as denoising, inpainting [28] and segmentation [5]. Models of natural images are also used to produce *representations* capturing features that can be employed in higher level tasks such as object recognition [10, 32, 14]. Learning good representations of images, as opposed to using engineered descriptors [21, 18, 3], is a crucial step towards building general systems that can easily adapt to different domains.

Devising models of natural images is challenging because images are continuous, high-dimensional and highly structured. Recent models have tried to capture high-order dependencies by using hierarchical models that extract highly non-linear representations of the input [13, 15]. In particular, *deep learning* methods construct hierarchies composed of multiple layers by greedily training each layer separately using unsupervised algorithms [10, 32, 17, 12]. These methods are very appealing because 1) they adapt to the data, 2) they recursively build hierarchies using unsupervised algorithms breaking up the difficult problem of learning hierarchical non-linear systems into a sequence of simpler learning tasks that use only unlabeled data, and 3) they have demonstrated good performance on a variety of domains, from handwritten character recognition to generic object recognition [10, 17, 12]. In this paper we propose a new module for deep learning that is specifically designed to represent natural images well. We start by giving some background and motivation for this approach.

1.1. Image Representations

The probabilistic model we are interested in has two sets of random variables, those that are observed and those that are latent and must be inferred. The former set consists of the pixels of an image patch, also called *visible* units; an example is given in fig. 1-A. The latter set consists of latent or *hidden* units that we would like to infer efficiently from the observation. The most efficient inference is achieved when the posterior distribution over the latent variables is factorial and simple to compute for each hidden unit, requiring no iterative time-consuming approximation. In order to understand how well the model captures the structure of the observed image, we can look at how well it reconstructs the image using the conditional distribution over the pixel values specified by its latent representation.

There are two main approaches. The most popular approach uses the hidden units to predict the mean intensity of each pixel independently from all the others. The conditional distribution is typically Gaussian with a mean depending on the configuration of the hiddens and a fixed diagonal covariance matrix. Methods such as sparse coding [22], probabilistic PCA [2], Factor Analysis [8] and the Gaussian RBM [4, 10, 1, 17, 16] use this approach. It is conceptually simple but it does not model the strong depen-

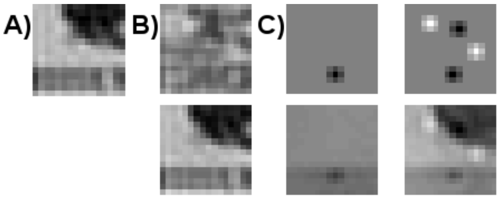

Figure 1. **A)** Input image patch. **B)** Reconstruction performed using only mean hiddens (top) and both mean and covariance hiddens (bottom) (that is multiplying the patch on the top by the image-specific covariance determined by the covariance hiddens that are inferred from the input image). **C)** Reconstructions produced by combining the correct image-specific covariance with the incorrect, hand-specified pixel intensities shown in the top row. Knowledge about pair-wise dependencies allows a blob of high or low intensity to be spread out over the appropriate region.

dencies that occur between neighboring pixels, and therefore, it fails to capture one of the most important properties of natural images.

The other approach employs hierarchical models [24, 13, 27] to capture these smoothness constraints. Pair-wise dependencies between pixels are modeled by using a zero-mean Gaussian distribution with an image-specific covariance that depends on the configuration of the hidden units. While able to capture smoothness between nearby pixels, these models lack information about the mean intensity and it is not obvious how to integrate that information and keep inference efficient at the same time.

The different nature of these two approaches suggests they might work well together providing a better representation of images. In this paper we propose a new probabilistic model that combines these approaches into a single framework. The model, dubbed *mcRBM*, has two sets of hidden units, one that represents the pixel intensities and another one that represents pair-wise dependencies between pixel intensities. The model is defined in terms of a joint Boltzmann distribution over the visibles and the hiddens with no connections between the hiddens, making inference very efficient since the hiddens are all independent given the visibles. Also, the hidden units are binary and they can be used as input to a standard binary Deep Belief Network (DBN) yielding a simple method to build a hierarchical model of natural images with many layers of non-linear features.

The intuition behind the proposed method is the following. Suppose the model knows about pair-wise correlations in the image. For instance, it knows that the pixels in the lower part of the image in fig. 1-A are strongly correlated. The model knows these pixels are likely to take the same value, but it does not know which one. Then, very noisy information about the values of the individual pixels in the lower part of the image would be sufficient to reconstruct the whole region quite well, since the model knows which values can be smoothed. We can repeat the same argument for the pixels in the top right corner and for those in the middle part of the image as well. Fig. 1-C illustrates this concept. Images on the top show mean intensities as provided to the model and the bottom row shows the corresponding reconstructions. Information about intensity is propagated over each region thanks to the pair-wise dependencies captured by the hidden units modeling the covariance. Fig. 1-B shows the same using the actual mean intensity produced by the mean hidden units (top). The reconstruction produced by the model using the whole set of hidden units is very close to the input image, as it can be seen in bottom part of fig. 1-B.

In sec. 2 we describe the model and its properties. Sec. 3 explains how the parameters of the model are learned. Finally, sec. 4 will demonstrate the performance of the model by learning features on natural image patches and using these features to recognize object categories in the CIFAR 10 dataset [16].

2. Model

First we describe how we can map real-valued images into binary features that capture only pair-wise dependencies between the pixels. Then, we augment the model by taking into account the predicted means of the pixel intensities.

2.1. Modeling the Covariance

The probabilistic model is defined in terms of an energy function E. We can derive the probability density function from E as $p(\mathbf{v}) \propto exp(-E(\mathbf{v}))$, where $\mathbf{v} \in R^D$ is the vectorized input image with D pixels. In order to capture pair-wise interactions between pixel values we only need to define E as a weighted sum of products between pairs of visible units. The easiest way to achieve this and to guarantee that the distribution is normalizable, is to define E as a positive linear combination of squared filter outputs because that makes the distribution Gaussian over the visibles. Let us denote with $C \in R^{D \times F}$ the matrix collecting F filters in its columns, then $E(\mathbf{v}) = -0.5 \sum_f P_{ff} (\sum_i C_{if} v_i)^2$ with P being a diagonal matrix with non-positive entries, defines a zero-mean Gaussian distribution over $\mathbf{v}$. We can also modulate or gate each term in the sum by multiplying it by a hidden unit state h_k^c. The more general form of the energy function becomes:

$$E^c(\mathbf{v}, \mathbf{h}^c) = -\frac{1}{2} \sum_{f=1}^{F} \sum_{k=1}^{N} P_{fk} h_k^c (\sum_{i=1}^{D} C_{if} v_i)^2 - \sum_{k=1}^{N} b_k^c h_k^c \quad (1)$$

where $P \in R^{F \times N}$ is a matrix with non-positive entries, N is the number of hidden units and $\mathbf{b}^c$ is a vector of biases. Each term in the first sum consists of a triplet of vari-

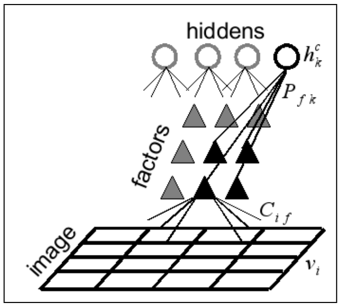

Figure 2. Toy illustration of the model. Factors (the triangles) compute the projection of the input image (whose pixels are denoted by v_i) with a set of filters (columns of matrix C). Their output is squared because each factor is connected twice to the same image with the same set of filters. The square outputs are sent to binary hidden units after projection with a second layer matrix (matrix P) that pools similar filters. Because the second layer matrix is non-positive the binary hidden units use their "off" states to represent abnormalities in the covariance structure of the data.

ables that are multiplied together: two visibles and one hidden. This is an example of a *third-order* Boltzmann Machine [29]. A given configuration of the hidden units $\mathbf{h}^c$, that we call *covariance* hiddens, specifies the interaction between pairs of visibles. The number of covariance matrices that the model can generate is exponential in the number of hidden units since the representation is binary and distributed. Unlike general higher-order Boltzmann Machines, the interaction between the visibles and the hiddens is *factorized* to reduce the number of parameters. The system can be represented as in fig. 2 where there are deterministic factors that are connected twice to the image with the same set of filters and once to the hidden units. We call this model *cRBM* because it is a Restricted Boltzmann Machine [4] that models the covariance structure of images.

Since the energy adds terms that have only one hidden unit the conditional distribution over the hiddens is factorial. The conditional distribution over the k-th hidden unit is:

$$p(h_k^c = 1 \mid \mathbf{v}) = \sigma(\frac{1}{2}\sum_{f=1}^{F} P_{fk}(\sum_{i=1}^{D} C_{if} v_i)^2 + b_k^c) \qquad (2)$$

where $\sigma(x) = 1/(1 + exp(-x))$ is the logistic function. The computation required for each hidden unit is consistent with biologically inspired models of low level vision where a *simple-cell* projects the input along a filter, and *complex-cells* pool many rectified simple cell outputs together to achieve a more invariant representation. The conditional distribution over the visibles is a zero-mean Gaussian distribution with an inverse covariance matrix that depends on $\mathbf{h}^c$:

$$\Sigma^{-1} = C\text{diag}(P\mathbf{h}^c)C' \qquad (3)$$

2.2. Modeling Mean and Covariance

In order to take into account the predicted mean intensities of the visible units we add an extra term to the energy function: $E(\mathbf{v}, \mathbf{h}^c, \mathbf{h}^m) = E^c(\mathbf{v}, \mathbf{h}^c) + E^m(\mathbf{v}, \mathbf{h}^m)$, where E^c is defined in eq.1, and E^m is defined as:

$$E^m(\mathbf{v}, \mathbf{h}^m) = -\sum_{j=1}^{M}\sum_{i=1}^{D} W_{ij} h_j^m v_i - \sum_{j=1}^{M} b_j^m h_j^m \qquad (4)$$

where there are M hidden units h_j^m, dubbed *mean hiddens*, with direct connections W_{ij} to the visibles and biases $\mathbf{b}^m$. The conditional distribution over the hiddens is:

$$p(h_j^m = 1 \mid \mathbf{v}) = \sigma(\sum_{i=1}^{D} W_{ij} v_i + b_j^m) \qquad (5)$$

while the conditional distribution over the visibles becomes:

$$p(\mathbf{v} \mid \mathbf{h}^c, \mathbf{h}^m) \sim N\left(\Sigma(\sum_{j=1}^{M} W_{ij} h_j^m),\ \Sigma\right) \qquad (6)$$

where Σ is given in eq. 3. The mean of the conditional over the visibles depends on both covariance and mean hidden units, and the conditional over both sets of hiddens remains factorial.

We can make the model more robust to large variations in the contrast of the input image by computing not the inner product but the cosine of the angle between input image and filters. We perform this normalization only in the energy function E^c. The overall energy becomes:

$$E(\mathbf{v}, \mathbf{h}^c, \mathbf{h}^m) = -\frac{1}{2}\sum_{f=1}^{F}\sum_{k=1}^{N} P_{fk} h_k^c (\sum_{i=1}^{D} \frac{C_{if}}{\|C_f\|}\frac{v_i}{\|\mathbf{v}\|})^2$$
$$-\sum_{k=1}^{N} b_k^c h_k^c + \frac{1}{2}\sum_{i=1}^{D} v_i^2 - \sum_{j=1}^{M}\sum_{i=1}^{D} W_{ij} h_j^m v_i - \sum_{j=1}^{M} b_j^m h_j^m \qquad (7)$$

where C_f is the f-th column (filter) of matrix C. This slight change makes the model more robust without affecting the efficiency of inference, but it further complicates the conditional distribution over the visibles, making it non-Gaussian [1]. Notice that we add a quadratic regularization term to the energy to guarantee normalization of the distribution over the whole space (since the covariance term would now be invariant to the norm of the input). We call this model *mcRBM* because it is a RBM that models both the mean and covariance structure of images.

[1] Indeed, we use the conditional distribution over the visibles only to produce the demos in fig. 1 and 6. We approximated that by simply rescaling the covariance of the Gaussian by the norm of the input image.

2.3. Analyzing the Model

We now give an intuition of how the model works; for ease of presentation we consider the formulation of the energy function without normalization of filters and input.

Since the hidden units are binary, they can be easily integrated out and we can compute the analytical form of the free energy F of a visible vector. The free energy is used to compute the marginal distribution over the visibles, $p(\mathbf{v}) \propto exp(-F(\mathbf{v}))$. Disregarding the normalization, the free energy is:

$$\begin{aligned} F(\mathbf{v}) \quad = \quad & -\sum_{k=1}^{N} \log(1 + e^{\frac{1}{2}\sum_f P_{fk}(\sum_i C_{if} v_i)^2 + b_k^c}) \\ & -\sum_{j=1}^{M} \log(1 + e^{\sum_i W_{ij} v_i + b_j^m}) \end{aligned} \tag{8}$$

The marginal distribution over the input is a product of $N + M$ experts. In particular, each (covariance) expert in the first sum is a mixture of a very broad Gaussian (a uniform distribution) and a more concentrated Gaussian (that depends on C and P). This is the simplest example of a Gaussian Scale Mixture [33] that more generally combines an infinite number of zero mean Gaussian distributions with different variances. As a result each term in the first sum corresponds to the energy of a fat-tailed distribution. The filter output is expected to be zero most of the times, with occasional large deviations from zero. We therefore expect the filter outputs to be sparsely distributed and when learning on natural images we should expect the filters to look like Gabor band-pass functions, which are well known to produce such sparse marginal distributions. In other words, each covariance expert represents a constraint that is almost always satisfied. When there is a large violation of smoothness in the direction of the corresponding filter then the constraint is turned off.

The covariance hidden units $\mathbf{h}^c$ must also be sparse. Assuming positive biases $\mathbf{b}^c$ and recalling that $P \leq 0$ and that the filter outputs are mostly zero, the covariance hiddens spend most of the time in the "on" state, and only rarely will they take the "off" state, see eq. 2. The role of the covariance hidden units can be better understood by rewriting the covariance energy function (without biases) as:

$$E^c(\mathbf{v}, \mathbf{h}^c) = -\frac{1}{2}\mathbf{v}'[\sum_{k=1}^{N} h_k^c (\sum_{f=1}^{F} P_{fk} C_f C_f')]\mathbf{v} \tag{9}$$

The strong violation of a constraint (i.e. a non-zero filter output) causes the corresponding covariance hidden unit to become zero removing a term from the inverse covariance matrix defined by the energy function. This term is a weighted sum of rank one matrices and its removal makes the resulting Gaussian distribution become less compressed along the direction of the violated constraint. As a result the model will relax the default smoothness assumption and allow the pixels on opposite sides of an edge to have very different values.

The interaction between covariance and mean hiddens can be understood by analyzing the demo in fig. 1. After training the model we used eq. 2 and 5 to compute the hiddens given the input image patch. At this point, the model knows the correlation between every pair of pixels; e.g. it knows that pixels in the upper right corner are strongly correlated. In fig. 1-B we show the reconstruction of the mean hiddens before and after multiplying by the covariance produced by the covariance hiddens (see eq. 6). The reconstruction of the mean hiddens is very blurry and blobby, but when combined with the covariance it becomes very accurate. Fig. 1-C shows the same when the reconstruction of the mean hiddens is substituted by simple blobs. Knowledge about a single pixel intensity is sufficient to fill out whole regions thanks to the dependencies extracted by the covariance hiddens.

3. Learning

Let us denote a generic parameter of the model with $\theta \in \{C, P, \mathbf{b}^c, W, \mathbf{b}^m\}$. We learn the parameters by stochastic gradient ascent in the log likelihood. We can write the likelihood in terms of the joint energy E or in terms of the free energy F, see eq. 8 taking into account also the normalization of input and filters in matrix C. Since exact sampling from the conditional over the visibles is hard, we opt for using the free energy. The update rule is:

$$\theta \leftarrow \theta + \eta\frac{\partial L}{\partial \theta}, \quad \text{with } \frac{\partial L}{\partial \theta} = < \frac{\partial F}{\partial \theta} >_{\text{model}} - < \frac{\partial F}{\partial \theta} >_{\text{data}} \tag{10}$$

where $< >$ denotes expectation. While it is straightforward to compute the value and the gradient of the free energy with respect to θ, it is intractable to compute the expectations over the model distribution. We approximate that with Contrastive Divergence [7] and Hybrid Monte Carlo [20]. With a similar set up to Hinton *et al.* [11] we approximate a sample from the model by running a dynamical simulation for 20 "leap-frog steps" starting at the current training sample and adjusting the step size to keep the rejection rate at 10%. To reach equilibrium it would be necessary to repeatedly discard the momentum, add a new random momentum, and then run for more leap-frog steps. For contrastive divergence learning however, we only need to add the random momentum once at the initial data point.

The learning algorithm loops over batches of training samples and: (1) it computes $\frac{\partial F}{\partial \theta}$ at the training samples, (2) it generates negative samples by running HMC for just one set of 20 leap-frog steps, (3) it computes $\frac{\partial F}{\partial \theta}$ at the negative samples, and (4) it updates the weights using eq. 10.

After training, inference is straightforward. The binary latent features are all conditionally independent given the observed visibles. The two sets of hiddens are found by applying eq. 2 (adding the normalization of the input) and eq. 5.

Once this model is trained, it can produce features to train a second stage binary RBM [10], for instance. In this case, we propagate real-valued probabilities instead of binary states. The process can be repeated for as many stages as desired and it will yield abstract representations of data good at capturing higher-order correlations.

4. Experiments

We first perform feature extraction on a dataset of natural image patches to gain understanding on how the model works. Then, we report results on the CIFAR 10 dataset. We recognized object categories by training a multinomial logistic classifier on the features produced by the model and achieved the best performance on this dataset to date.

In all of the experiments, images were pre-processed by PCA whitening retaining 99% of the variance. The algorithm was optimized by using stochastic gradient ascent over batches of 128 samples, with a learning rate set to 0.15 for matrix C, to 0.01 for W, and 0.001 for matrix P and the biases. Learning rates were slowly annealed during training. HMC was initialized at an observation with a random momentum followed by 20 leap-frog steps. All parameters were randomly initialized to small values. The columns of matrix P were normalized to unit L1 norm, similarly to Osindero *et al*. [24]. We always initialized P with the negative identity or by using a topography over the filter outputs of matrix C. After the filters converged, we started updating P. After every update of P we set to zero those entries that were positive in order to satisfy the non-positivity constraint.

4.1. Learning from Natural Images

We generated a dataset of 500,000 color images by picking, at random locations, patches of size 16x16 pixels from images of the Berkeley segmentation dataset [2]. We first trained a model with 256 factors, 256 covariance hiddens and 100 mean hiddens. P was initialized to the negative identity. Fig. 3 shows the learned filters in matrix C. Most filters learn to be balanced in the R, G, and B channels. They could be described by localized, orientation-tuned Gabor functions. The colored filters generally have lower spatial frequency. The filters capturing the mean intensities (columns of matrix W) are shown in fig. 4. These features are not localized, showing broad variations of intensity and color. In order to visualize matrix P, we look at the filters that are most strongly connected to each hidden unit. Fig. 5 shows a subset of those and it demonstrates that each hidden unit pools filters that have patterns in similar location, orientation, scale and with similar color. Each covariance hidden unit is not only invariant to the sign of the input, because filter outputs are squared, but also is invariant to more com-

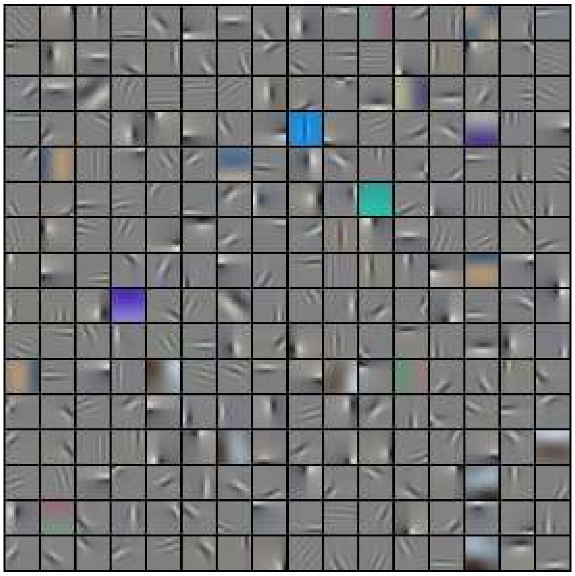

Figure 3. 256 filters (columns of matrix C) learned on color image patches of size 16x16 pixels (best viewed in color).

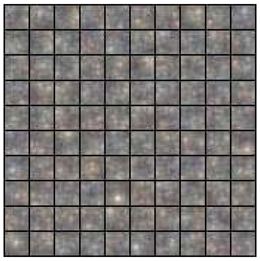

Figure 4. 100 mean filters learned in matrix W (best viewed in color).

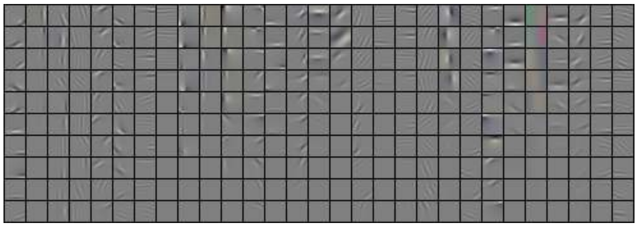

Figure 5. Visualization of the pooling achieved by matrix P. Each column shows the filters in C that are most strongly connected to a covariance hidden unit (showing only a random subset of hiddens and up to 10 filters). Filters are sorted according to the strength of their connection and the contrast is also set proportional to that strength.

[2] http://www.cs.berkeley.edu/projects/vision/grouping/segbench/

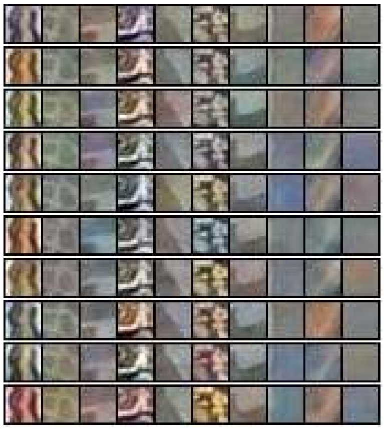

Figure 6. Stochastic reconstructions using the conditional distribution over the visibles. Samples along the columns share the same configuration of hidden units (best viewed in color).

plex local distortions since a hidden unit turns off whenever any filter among those pooled becomes highly active.

The invariance properties of the latent representation can be investigated also by looking at stochastic reconstructions from the conditional distribution over the visibles. Given some training samples, we infer both sets of hidden units. Then, we generate samples from the conditional distribution over the visibles clamping the hiddens to the inferred state. Fig. 6 shows samples generated using this procedure. Different samples have slightly different color, different position of edges and different contrast. All samples in a column are associated to the same configuration of hiddens. The hidden representation exhibits robustness to small distortions.

mcRBM is a probabilistic model of images and it can also be assessed by the quality of its samples. We trained a larger model with 1024 factors, 1024 covariance hiddens and 576 mean hiddens. After training, we run HMC for a very long time starting from a random image. Fig. 7 shows that mcRBM is actually able to generate samples that look remarkably similar to natural images, and qualitatively superior to those generated by a Gaussian RBM (modeling only the mean intensity) and by a cRBM (modeling only pair-wise dependencies between pixels). Not only are there noisy texture patches but also uniform patches and patches with long elongated structures that cover the whole image.

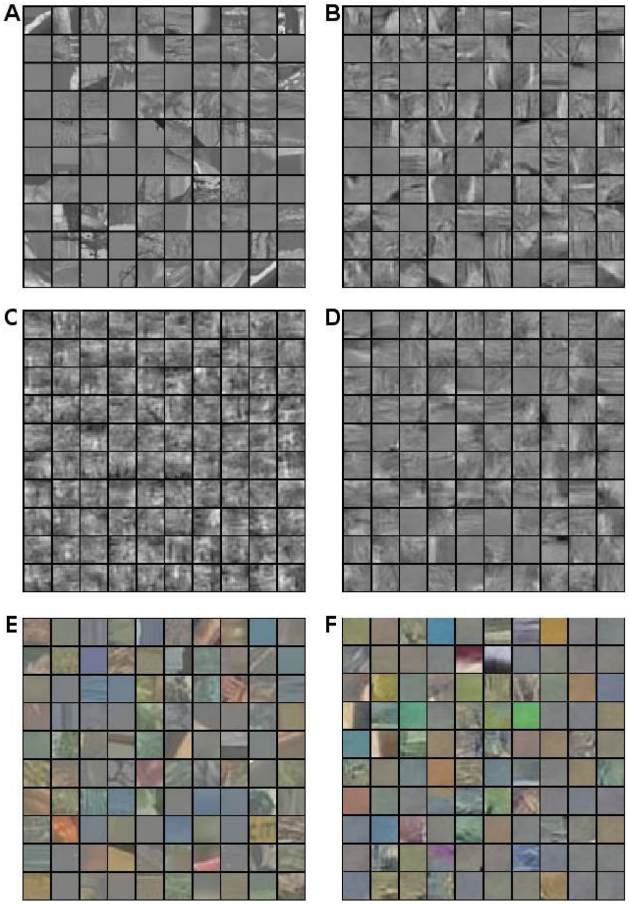

Figure 7. A) Gray-scale patches used during training. B), C) and D) Samples drawn from mcRBM, GRBM and cRBM, respectively. E) Color patches used during training. F) Samples drawn from mcRBM trained on color patches.

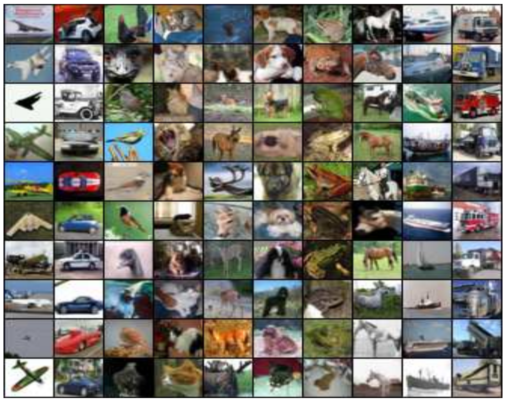

Figure 8. Example of images in the CIFAR 10 dataset. Each column shows samples belonging to the same category.

4.2. Object Recognition on CIFAR 10

The CIFAR 10 dataset [16] is a hand-labeled subset of a much larger dataset of 80 million tiny images [30], see fig. 8. These images were downloaded from the web and down-sampled to a very low resolution, just 32x32 pixels. The CIFAR 10 subset has ten object categories, namely airplane, car, bird, cat, deer, dog, frog, horse, ship, and truck. The training set has 5000 samples per class, the test set has 1000 samples per class. The low resolution and

Method	Accuracy %
1) mean (GRBM): *11025*	59.7 (72.2)
2) cRBM (225 factors): *11025*	63.6 (83.9)
3) cRBM (900 factors): *11025*	64.7 (80.2)
4) mcRBM: *11025*	68.2 (83.1)
5) mcRBM-DBN (*11025-8192)*	70.7 (85.4)
6) mcRBM-DBN (*11025-8192-8192)*	**71.0** (83.6)
7) mcRBM-DBN (*11025-8192-4096-1024-384)*	59.8 (62.0)

Table 1. Test and training (in parenthesis) recognition accuracy on the CIFAR 10 dataset varying the components in the model (using only mean, only covariance and both), varying the depth and feature dimensionality. The numbers in italics are the feature dimensionality at each stage. The first stage maps the input into a 11025 dimensional representation.

Method	Accuracy %
384 dimens. GIST	54.7
10,000 linear random projections	36.0
10K GRBM(*), 1 layer, ZCA'd images	59.6
10K GRBM(*), 1 layer	63.8
10K GRBM(*), 1layer with fine-tuning	64.8
10K GRBM-DBN(*), 2 layers	56.6
11025 mcRBM 1 layer, PCA'd images	**68.2**
8192 mcRBM-DBN, 3 layers, PCA'd images	**71.0**
384 mcRBM-DBN, 5 layers, PCA'd images	**59.8**

Table 2. Test recognition accuracy on the CIFAR 10 dataset produced by different methods. Features are fed to a multinomial logistic regression classifier for recognition. Results marked by (*) are obtained from [16], our model is denoted by mcRBM.

extreme variability make recognition very difficult and a traditional method based on features extracted at interest-points is unlikely to work well. Moreover, extracting features from such images using carefully engineered descriptors like SIFT [18] or GIST [21] is also likely to be suboptimal since these descriptors were designed to work well on higher resolution images. Previous work on this dataset has used GIST [31] and Gaussian RBM's [16].

We use the following protocol. We train mcRBM on 8x8 color image patches sampled at random locations, and then we apply the algorithm to extract features convolutionally over the whole 32x32 image by extracting features on a 7x7 regularly spaced grid (stepping every 4 pixels). Then, we use a multinomial logistic regression classifier to recognize the object category in the image. Since our model is unsupervised, we train it on a set of two million images from the TINY dataset that does not overlap with the labeled CIFAR 10 subset in order to further improve generalization [9, 26, 25]. In the default set up we learn all parameters of the model, we use 81 filters in W to encode the mean, 576 filters in C to encode covariance constraints and we pool these filters into 144 hidden units through matrix P. P is initialized with a two-dimensional topography that takes 3x3 neighborhoods of filters with a stride equal to 2. In total, at each location we extract 144+81=225 features. Therefore, we represent a 32x32 image with a 225x7x7=11025 dimensional descriptor.

Table 1 shows some interesting comparisons. First, we assess whether it is best to model just the mean intensity, or just the covariance or both in 1), 2) and 4). In order to make a fair comparison we used the same feature dimensionality. The covariance part of the model produces features that are more discriminative, but modelling both mean and covariance further improves generalization. In 2) and 3) we show that increasing the number of filters in C while keeping the same feature dimensionality (by pooling more with matrix P) also improves the performance. We can allow for a large number of features as long as we pool later into a more compact and invariant representation. Entries 4), 5) and 6) show that adding an extra layer on the top (by training a binary RBM on the 11025 dimensional feature) improves generalization. Using three stages and 8192 features we achieved the best performance of **71.0%**.

We also compared to the more compact 384 dimensional representation produced by GIST and found that our features are more discriminative, as shown in table 2. Previous results using GRBM's [16] reported an accuracy of 59.6% using whitened data while we achieve 68.2%. Their result improved to 64.8% by using unprocessed data and by fine-tuning, but it did not improve by using a deeper model. Our performance improves by adding other layers showing that these features are more suitable for use in a DBN.

5. Related Work

Osindero and Hinton [23] use a type of RBM in which the binary visible units have direct pairwise interactions that are learned, but these interactions are not modulated by the activities of hidden units: The hidden units only determine the effective biases of the visible units. The Product of Student's t (PoT) model [24] is also closely related to ours because it describes and modulates pair-wise interactions between real-valued pixels with hidden units. However, PoT does not model mean intensity and it maps real valued images into Gamma distributed hidden units that are more difficult to use in deep learning algorithms. Heess *et al.* [6] proposed an extension of PoT to make the energy function more flexible and, in particular, to include the mean intensity of pixels. However, in their model the mean does not depend on hidden units but is a fixed parameter subject to learning. Our preliminary version of this work [27] did not take into account the mean intensity of the image and did not include the normalization of the input in the energy function. Finally, factored higher-order Boltzmann Machines were first introduced by Memisevic and Hinton [19] to represent conditional distributions not joint distributions as in this work.

6. Conclusions

Motivated by the need to capture both the intensities and the relationships between the intensities in an image patch, we proposed a new model, mcRBM, which maps real-valued images into binary representations. Our experiments demonstrate that these binary features achieve the best reported accuracy on the challenging CIFAR 10 dataset. Unlike other models, mcRBM can also generate very realistic samples that have coherent and elongated structures.

In the future, we plan to extend mcRBM to large images by using a Field of mcRBMs in which the tied weights are all trained together. We also plan to use mcRBM for other tasks, such as segmentation and denoising.

Acknowledgements

The authors thank I. Murray, H. Larochelle, I. Sutskever and R. Adams for their suggestions, V. Mnih for providing the CUDAMAT package for GPU, and NSERC, CFI & CIFAR for funding.

References

[1] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle. Greedy layer-wise training of deep networks. In *NIPS*, 2007. 1

[2] C. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society, Series B*, 61:611–622, 1999. 1

[3] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 1

[4] Y. Freund and D. Haussler. Unsupervised learning of distributions on binary vectors using two layer networks. Technical report, University of Calif. Santa Cruz, 1994. 1, 3

[5] X. He, R. Zemel, and D. Ray. Learning and incorporating top-down cues in image segmentation. In *ECCV*, 2006. 1

[6] N. Heess, C. Williams, and G. Hinton. Learning generative texture models with extended fields-of-experts. In *BMCV*, 2009. 7

[7] G. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14:1771–1800, 2002. 4

[8] G. Hinton, P. Dayan, and M. Revow. Modeling the manifolds of images of handwritten digits. *IEEE Transactions on Neural Networks*, 8:65–74, 1997. 1

[9] G. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural Comp.*, 18:1527–1554, 2006. 7

[10] G. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006. 1, 5

[11] G. E. Hinton, S. Osindero, M. Welling, and Y. Teh. Unsupervised discovery of non-linear structure using contrastive backpropagation. *Cognitive Science*, 30:725–731, 2006. 4

[12] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun. What is the best multi-stage architecture for object recognition? In *ICCV*, 2009. 1

[13] Y. Karklin and M. Lewicki. Emergence of complex cell properties by learning to generalize in natural scenes. *Nature*, 457:83–86, 2009. 1, 2

[14] K. Kavukcuoglu, M. Ranzato, R. Fergus, and Y. LeCun. Learning invariant features through topographic filter maps. In *CVPR*, 2009. 1

[15] U. Koster and A. Hyvarinen. A two-layer ica-like model estimated by score matching. In *ICANN*, 2007. 1

[16] A. Krizhevsky. Learning multiple layers of features from tiny images, 2009. MSc Thesis, Dept. of Comp. Science, Univ. of Toronto. 1, 2, 6, 7

[17] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proc. ICML*, 2009. 1

[18] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004. 1, 7

[19] R. Memisevic and G. Hinton. Learning to represent spatial transformations with factored higher-order boltzmann machines. *Neural Comp.*, 2010. 7

[20] R. Neal. *Bayesian learning for neural networks*. Springer-Verlag, 1996. 4

[21] A. Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *IJCV*, 42:145–175, 2001. 1, 7

[22] B. A. Olshausen and D. J. Field. Sparse coding with an overcomplete basis set: a strategy employed by v1? *Vision Research*, 37:3311–3325, 1997. 1

[23] S. Osindero and G. E. Hinton. Modeling image patches with a directed hierarchy of markov random fields. In *NIPS*, 2008. 7

[24] S. Osindero, M. Welling, and G. E. Hinton. Topographic product models applied to natural scene statistics. *Neural Comp.*, 18:344–381, 2006. 2, 5, 7

[25] R. Raina, A. Battle, H. Lee, B. Packer, and A. Ng. Self-taught learning: Transfer learning from unlabeled data. In *ICML*, 2007. 7

[26] M. Ranzato, F. Huang, Y. Boureau, and Y. LeCun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *CVPR'07*, 2007. 7

[27] M. Ranzato, A. Krizhevsky, and G. Hinton. Factored 3-way restricted boltzmann machines for modeling natural images. In *AISTATS*, 2010. 2, 7

[28] S. Roth and M. Black. Fields of experts: A framework for learning image priors. In *CVPR*, 2005. 1

[29] T. Sejnowski. Higher-order boltzmann machines. In *AIP Conf. proc., Neural networks for computing*, 1986. 3

[30] A. Torralba, R. Fergus, and W. Freeman. 80 million tiny images: a large dataset for non-parametric object and scene recognition. *PAMI*, 30:1958–1970, 2008. 6

[31] A. Torralba, R. Fergus, and Y. Weiss. Small codes and large databases for recognition. In *CVPR*, 2008. 7

[32] P. Vincent, H. Larochelle, Y. Bengio, and P. Manzagol. Extracting and composing robust features with denoising autoencoders. In *ICML*, 2008. 1

[33] M. Wainwright and E. Simoncelli. Scale mixtures of gaussians and the statistics of natural images. In *NIPS*, 2000. 4