Learning a generative model of images by factoring appearance and shape

Nicolas Le Roux[1], **Nicolas Heess**[2], **Jamie Shotton**[1], **John Winn**[1]

[1]Microsoft Research Cambridge

[2]University of Edinburgh

Keywords: generative model of images, occlusion, segmentation, deep networks

Abstract

Computer vision has grown tremendously in the last two decades. Despite all efforts, existing attempts at matching parts of the human visual system's extraordinary ability to understand visual scenes lack either scope or power. By combining the advantages of general low-level generative models and powerful layer-based and hierarchical models, this work aims at being a first step towards richer, more flexible models of images. After comparing various types of RBMs able to model continuous-valued data, we introduce our basic model, the masked RBM, which explicitly models occlusion boundaries in image patches by factoring the appearance of any patch region from its shape. We then propose a generative model of larger images using a field of such RBMs. Finally, we discuss how masked RBMs could be stacked to form a deep model able to generate more complicated structures and suitable for various tasks such as segmentation or object recognition.

1 Introduction

Despite much progress in the field of computer vision in recent years, interpreting and modeling the bewildering structure of natural images remains a challenging problem. The limitations of even the most advanced systems become strikingly obvious when contrasted with the ease, flexibility, and robustness with which the human visual system analyzes and interprets an image. Computer vision is a problem domain where the structure that needs to be represented is complex and strongly task dependent, and the input data is often highly ambiguous. Against this background, we believe that rich generative models are necessary to extract an accurate and meaningful representation of the world, detailed enough to make them suitable for the wide range of visual tasks. This work is a first step towards building such a general-purpose generative model able to perform varied high-level tasks on natural images. The model integrates concepts from computer vision that combine some very general knowledge about the structure of our visual world with ideas from "deep" unsupervised learning. In particular it draws on ideas such as:

- the separation of shape and appearance and the explicit treatment of occlusions;
- a generic, learned, model of shapes and appearances;
- the unsupervised training of a generative model on a large database, exploiting graphical models that allow for efficient inference and learning;
- the modeling of large images using a field of more local experts;
- the potential for a hierarchical latent representation of objects.

Some of these ideas have been explored independently of each other, and in models that focused on particular aspects of images or that were applied to very limited (e.g. category specific) datasets. Here we demonstrate how these techniques, in combination, give rise to a promising model of generic natural images.

One premise of the work described in the remainder of this paper is that generative models hold important advantages in computer vision. Their most obvious advantage over discriminative methods is perhaps that they are more amenable to unsupervised learning, which seems of crucial importance in a domain where labeled training data is

often expensive while unlabeled data is nowadays easy to obtain. Equally important, however, is that in vision we are rarely interested in solving a single "task" such as object classification. Instead we typically need to extract information about different aspects of an image and at different levels of abstraction, e.g. recognizing whether an object is present, identifying its position and those of its parts, and separating pixels belonging to the object from the background or occluding objects (segmentation), etc. Many lower-level tasks, such as segmentation, are not even well defined without reference to more abstract structure (e.g. the object or part to be segmented), and information in natural images, especially when it is low-level and local, is often highly ambiguous. These considerations strongly suggests that we need a model that is able to represent and learn a rich prior of image structure at many different levels of abstraction, and that also allows to efficiently combine bottom-up (from the data) with top-down (from the prior) information during inference. Probabilistic, generative models naturally offer the appropriate framework for doing such inference. Furthermore, unlike in the discriminative case, they are trained not with respect to a particular, task-specific, label (which in most cases provides very little information about the complex structure present in an image) but rather to represent the data efficiently. This makes it much more likely that the required rich prior can ultimately be learned, especially if a suitable, e.g. hierarchical model structure is assumed. In the following we will briefly review the most closely related works even though such a review will necessarily have to remain incomplete.

Some generative models can extract information about shape and appearance, illumination, occlusion and other factors of variation in an unsupervised manner (Frey and Jojic, 2003; Williams and Titsias, 2004; Kannan *et al.*, 2005; Winn and Jojic, 2005; Kannan *et al.*, 2006). Though these models have successfully been applied to sets of relatively homogeneous images, e.g. images of particular object classes or movies of a small number of objects, they have limited scope and are typically not suitable for more heterogeneous data, let alone generic natural images.

Generic image structure is the domain of models such as the sparse coding approach by Olshausen & Field (Olshausen and Field, 1996; Lewicki and Olshausen, 1999; Hyvärinen *et al.*, 2001; Karklin and Lewicki, 2009) or the more recent work, broadly referred to as deep learning architectures (Osindero and Hinton, 2008; Lee *et al.*, 2008). Unlike the models in the previous category, these models of *generic* im-

age structure have very little "built-in" knowledge about the formation of natural images and are trained on large unlabeled image databases. In particular, for the second group of models the hope is that by learning increasingly deep (i.e. multi-layered) representations of natural images these models will capture structures of increasing complexity and at larger scales. Although this line of work has produced interesting results, so far the models are typically limited to small image patches (with some exceptions, see for instance Lee *et al.* (2009) and Raina *et al.* (2009)). Furthermore, most models so far, including hierarchical ones, appear to learn only very simple, low-level properties of natural images and are far from learning more abstract, higher-level concepts, suggesting that these models might still be too limited to capture the wealth of structure in natural images.

The question as to what kind of models are suitable for modeling the very different types of structure occurring in natural images has featured prominently in the work of Zhu and his coworkers (Guo *et al.*, 2003; Tu *et al.*, 2005; Zhu and Mumford, 2006; Guo *et al.*, 2007). Recently, they have proposed a comprehensive generative model which combines sub-models of different types for capturing the different types of structure occurring in natural images at different levels of abstraction and scale, ranging from low-level structures such as image textures to high-level part-based representations of objects and ultimately full visual scenes. However, many aspects of this model are handcrafted and it fails to leverage one of the potential advantages of generative models in that unsupervised learning seems extremely difficult if not impossible.

Last, image models are often formulated in terms of tree structured hierarchies, where each unit in the lower layer (representing, for instance, a pixel or a part) is explained by exactly one higher level unit. One important insight that has arisen from probabilistic work on image modeling such as the Dynamic Trees (Williams and Adams, 1999; Storkey and Williams, 2003), and also the credibility network model (Hinton *et al.*, 2000), is the notion that such a hierarchy needs to be flexible and allowed to vary in structure so as to match the underlying dependencies present in any particular image. However, these methods still fall short of being able to capture the complexity of natural images: for example, Dynamic Trees do not impose a depth ordering or learn an explicit shape model as a prior over tree structures.

In the light of all these works, we aim at providing a unified probabilistic framework able to deal with generic, large images in an efficient manner, both from a representation and an inference point of view.

The base component of our model will be the Restricted Boltzmann Machine (Smolensky, 1986; Freund and Haussler, 1994), which is a Boltzmann Machine (Ackley *et al.*, 1985) restricted to have bipartite connectivity. Section 2 presents and compares various RBMs able to model continuous-valued data, which will prove useful when we will model appearances of objects. Section 3 presents the masked RBM, which extends the already rich modeling capacity of an RBM with a depth-ordered segmentation model. The masked RBM represents the shape and appearance of image regions separately, and explicitly reasons about occlusion. The shape of objects is modeled by another RBM, introduced in section 4. This opens up new application domains (such as image segmentation and inpainting), and, importantly, leads to a much more efficient representation of image structure than standard RBMs, that can be learned in a fully unsupervised manner from training images. Despite its complexity and power, our model allows for efficient approximate inference and learning. Section 5 is a thorough evaluation of this model's quality using both toy data and natural image patches, demonstrating how explicit incorporation of knowledge about natural images formation considerably increases the efficiency of the learned representation.

We then move from image patches to large ones by introducing the field of masked RBMs in section 6, leveraging the modeling power we obtained at the patch level, before concluding in section 7.

Finally, as future work, we propose in section 8 a hierarchical formulation of the basic model which gives rise to a flexible, reconfigurable tree-structured representation that would allow us to learn image structures at different scales and levels of abstraction.

2 Binary and continuous-valued RBMs

In this section, we first introduce the standard RBM, defined over binary variables, before presenting several RBMs able to model continuous-valued data.

2.1 The Binary RBM

A binary RBM with n hidden units is a parametric model of the joint distribution between binary hidden variables h_j (explanatory factors, collected in vector $\mathbf{h}$) and binary observed variables v_i (the observed data, collected in vector $\mathbf{v}$), of the form

$$\log P(\mathbf{v}, \mathbf{h}) = \mathbf{v}^T W \mathbf{h} + \mathbf{b}^T \mathbf{v} + \mathbf{c}^T \mathbf{h} - \log Z^1 \ , \tag{1}$$

with parameters $\theta = (W, \mathbf{b}, \mathbf{c})$ and $v_i, h_j \in \{0, 1\}$ (Z is the normalizing constant). One can show that conditional distributions $P(\mathbf{v}|\mathbf{h})$ and $P(\mathbf{h}|\mathbf{v})$ are factorial and thus easy to sample from (Hinton, 2002). Although the marginal distribution $P(\mathbf{v})$ is not tractable, it can be easily computed up to a normalizing constant. The bipartite structure of an RBM allows both inference and learning to be performed efficiently using Gibbs sampling (Hinton *et al.*, 2006).

2.2 Modeling continuous values with an RBM

Since we are building a generative model of RGB images, we will need to use generative models of (potentially bounded) real-valued vectors of the red, green and blue channel values. Surprisingly, little work has been done on designing efficient RBMs for real-valued data.

The general foundations for using RBMs to model distributions in the exponential family were laid in Welling *et al.* (2005), where one particular instantiation of this family was investigated for modeling discrete data using continuous latent variables. To date, using other members of this family to learn data variance has not been explored.

Some authors have used RBMs in the context of continuous values, using either a truncated exponential (Larochelle *et al.*, 2007) or Gaussians with fixed variance (Freund and Haussler, 1994; Lee *et al.*, 2008). In none of these cases is the variance learned. In the case of the truncated exponential, even though the variance does depend on the parameters, it is a deterministic function of the mean and cannot be separately optimised; we will thus refer to this model as having 'fixed' variance.

We now present several kinds of RBMs able to model continuous-valued data.

[1] Throughout the paper, we slightly abuse notation and use the variable Z for all partition functions, although they depend on the energy function.

2.3 Truncated exponential

The use of the truncated exponential with an RBM is a direct extension of the original formulation to continuous values. The energy functions remains identical:

$$E(\mathbf{v}, \mathbf{h}) = -\mathbf{v}^T W \mathbf{h} - \mathbf{b}^T \mathbf{v} - \mathbf{c}^T \mathbf{h} \tag{2}$$

but $\mathbf{v}$ may now take any value in $[0, 1]$. The conditional $P(\mathbf{v}|\mathbf{h})$ is a truncated exponential.

2.4 Gaussian RBM with fixed variance

Gaussian RBMs have already been studied (Freund and Haussler, 1994) and used (Lee *et al.*, 2008), but always in the context of a fixed variance. The energy function is of the form

$$E(\mathbf{v}, \mathbf{h}) = \frac{1}{2\sigma^2} \mathbf{e}^T \mathbf{v}^2 - \frac{1}{\sigma^2} \left(\mathbf{b}^T \mathbf{v} + \mathbf{c}^T \mathbf{h} + \mathbf{v}^T W \mathbf{h} \right) \tag{3}$$

where $\mathbf{v}^2$ is the vector whose i-th element is v_i^2 and $\mathbf{e} = [1, 1, \ldots, 1]^T$. This model is restricted to be a mixture of isotropic gaussians.

Choosing a fixed variance to use with this model is problematic: large variances makes training very noisy, whilst small variances cause training to get stuck in local maxima. A heuristic approach exists, which aims at avoiding the problems of a large fixed variance by using the mean of $P(\mathbf{v}|\mathbf{h})$, rather than a sample from it, during training. We will show the results obtained with the fixed variance model trained normally (*Gaussian - Fixed*) and trained using this heuristic (*Gaussian - Heuristic*).

2.5 Gaussian RBM with learned variance

We now present an extension of the Gaussian RBM model which allows for the modeling of the variance. We consider two similar models: the first uses the same hidden units to model both the mean and the precision (*Gaussian - Joint*), whilst the second uses different sets of hidden units for each (*Gaussian - Separate*).

2.5.1 Joint modeling of mean and precision

The energy function for this model represents the mean and precision jointly using a common set of hidden units:

$$E(\mathbf{v}, \mathbf{h}) = -\mathbf{v}^T W^{\mathrm{m}} \mathbf{h} - \left(\mathbf{v}^2\right)^T W^{\mathrm{p}} \mathbf{h} - \mathbf{v}^T \mathbf{b}^{\mathrm{m}} - \left(\mathbf{v}^2\right)^T \mathbf{b}^{\mathrm{p}} - \mathbf{c}^T \mathbf{h} \tag{4}$$

Denoting precision $\Lambda = -2(W^{\mathrm{p}} \mathbf{h} + \mathbf{b}^{\mathrm{p}})$, we have

$$P(v_i|\mathbf{h}) \sim \mathcal{N}\left(\frac{W^{\mathrm{m}}_{i,:}\mathbf{h} + b^{\mathrm{m}}_i}{\Lambda_i}, \frac{1}{\Lambda_i}\right). \tag{5}$$

In this model, the biases $\mathbf{b}^{\mathrm{p}}$ and weights W^{p} are forced to be negative.

2.5.2 Separate modeling of mean and precision

Here, the energy function uses one set of hidden units $\mathbf{h}^{\mathrm{m}}$ to model the mean, and a separate set of hidden units $\mathbf{h}^{\mathrm{p}}$ to model the precision:

$$E(\mathbf{v}, \mathbf{h}^{\mathrm{m}}, \mathbf{h}^{\mathrm{p}}) = -\mathbf{v}^T W^{\mathrm{m}} \mathbf{h}^{\mathrm{m}} - \left(\mathbf{v}^2\right)^T W^{\mathrm{p}} \mathbf{h}^{\mathrm{p}} - \mathbf{v}^T \mathbf{b}^{\mathrm{m}} - \left(\mathbf{v}^2\right)^T \mathbf{b}^{\mathrm{p}} - \left(\mathbf{c}^{\mathrm{m}}\right)^T \mathbf{h}^{\mathrm{m}} - \left(\mathbf{c}^{\mathrm{p}}\right)^T \mathbf{h}^{\mathrm{p}} \tag{6}$$

Denoting $\Lambda = -2(W^{\mathrm{p}} \mathbf{h}^{\mathrm{p}} + \mathbf{b}^{\mathrm{p}})$, we now have

$$P(v_i|\mathbf{h}^{\mathrm{m}}, \mathbf{h}^{\mathrm{p}}) \sim \mathcal{N}\left(\frac{W^{\mathrm{m}}_{i,:}\mathbf{h}^{\mathrm{m}} + b^{\mathrm{m}}_i}{\Lambda_i}, \frac{1}{\Lambda_i}\right). \tag{7}$$

In this model, the biases $\mathbf{b}^{\mathrm{p}}$ and weights W^{p} are forced to be negative.

2.6 Beta RBM

In the Beta RBM, the conditional distributions $P(\mathbf{v}|\mathbf{h})$ are Beta distributions whose means and variances are learnt during training.

If we were to simply apply the formula given by Welling *et al.* (2005), the energy function of the Beta RBM would be

$$\begin{aligned} E(\mathbf{v}, \mathbf{h}) \;\; = \;\; & -\log(\mathbf{v})^T W \mathbf{h} - \log(\mathbf{e} - \mathbf{v})^T U \mathbf{h} \\ & + \mathbf{e}^T \log(\mathbf{v}) + \mathbf{e}^T \log(\mathbf{e} - \mathbf{v}) - \mathbf{c}^T \mathbf{h} \,. \end{aligned} \tag{8}$$

In this formulation, each expert is a mixture of a uniform and a Beta distribution. Unfortunately, training such an RBM proved very difficult as turning a hidden unit on

could only increase the precision of the conditional distribution. Furthermore, there is no easy way of enforcing the positivity constraint on the parameters of the Beta distributions (enforcing all the elements of $\mathbf{a}$, $\mathbf{b}$, W and U to be positive resulted in too hard a constraint).

We therefore modified the energy so that each expert is a mixture of two Beta distributions. By doing so, we symmetrize the hidden units and we can have weaker constraints on the parameters while still retaining valid distributions. The new, modified energy function is then:

$$\begin{aligned} E(\mathbf{v},\mathbf{h}) &= -\log(\mathbf{v})^T W_1 \mathbf{h} - \log(\mathbf{v})^T W_2 (\mathbf{e}-\mathbf{h}) \\ &\quad -\log(\mathbf{e}-\mathbf{v})^T U_1 \mathbf{h} - \log(1-\mathbf{v})^T U_2 (\mathbf{e}-\mathbf{h}) \\ &\quad +\log(\mathbf{v}) + \log(\mathbf{e}-\mathbf{v}) - \mathbf{c}^T \mathbf{h} \,. \end{aligned} \tag{9}$$

with the elements of W_1, W_2, U_1 and U_2 restricted to be positive (note that we do not have the visible biases $\mathbf{a}$ and $\mathbf{b}$ anymore as these may be included in the weight matrices). As Beta distributions treat the boundary values (0 and 1) differently than the others, we extended their range to $[-\lambda, 1+\lambda]$ with $\lambda = \left(\frac{\sqrt{5}-1}{2}\right)$ [2].

2.7 Assessment of the quality of each RBM

To choose the most appropriate RBM for the real-valued red, blue and green channels, we compared all these models on natural image patches (of size 16 by 16), using three quantitative metrics: the reconstruction RMSE, the reconstruction log-likelihood and the imputation accuracy. The experiments were led on patches which were not seen during training.

2.7.1 Experimental setup

All models were trained on a training set of $383,300$ color image patches of size 16×16. Patches were extracted on a regular 16×16 grid from images from three different object recognition data sets: Pascal VOC, MSR Cambridge and the INRIA horse data

[2] $\lambda = \frac{\sqrt{5}-1}{2}$ has the properties that $\log(\lambda) = -\log(1+\lambda)$ and $\log(1+\lambda) - \log(\lambda) \approx 1$. The first property ensures that the range of inputs to the hidden units is symmetric around 0 and the second property ensures that $\log(\mathbf{v}+\lambda)$, $\log(1+\lambda-\mathbf{v})$ and $\mathbf{h}$ are approximately of the same amplitude.

set.[3] Red, green and blue color channels are concatenated, so that each model has 768 visible units. Where necessary an appropriately sized validation set was used.

We trained the model using gradient descent with persistent contrastive divergence (Tieleman, 2008) and batches of size 20. We used a small weight decay, and decreased the learning rate every epoch (one run through all training patches), dividing each epoch into batches.

The hyperparameters were not treated equally:

- The weight decay and decrease constant were manually fixed to $.0002$ and $.001$, respectively.
- The learning rate was optimized using the validation set, taking the learning rate that gives the best log-likelihood of the data given the inferred latent variables after one epoch.
- In the case of the Beta RBM, to get an idea of the effect of parameter λ we tried three different values of λ for the case of 256 hidden units. We decided beforehand to report for 512 and 1024 hidden units only the results for $\lambda = \frac{\sqrt{5}-1}{2}$.

Once the optimal learning rate was found, we trained each model for 20 epochs, in batches of size 50 patches. Models were trained for three different sizes of the hidden layer: 256, 512 and 1024 hidden units.

2.7.2 Reconstruction RMSE

This experiment is used to determine the ability of each RBM to correctly model the mean of the data. Reconstruction is performed as follows. Given a test patch $\mathbf{v}_{\text{test}}$, we sample a configuration of the hidden states $\mathbf{h}^\star$ from the conditional distribution $P(\mathbf{h}|\mathbf{v}_{\text{test}})$. Given this configuration $\mathbf{h}^\star$, we compute the average value of the visible states $E[P(\mathbf{v}|\mathbf{h}^\star)]$. This is called a *mean reconstruction* of the test patch. Note that this is not the true average reconstruction since we only consider one configuration

[3] Available from `http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2008/`, `http://research.microsoft.com/vision/cambridge/recognition/`, and `http://lear.inrialpes.fr/data` respectively.

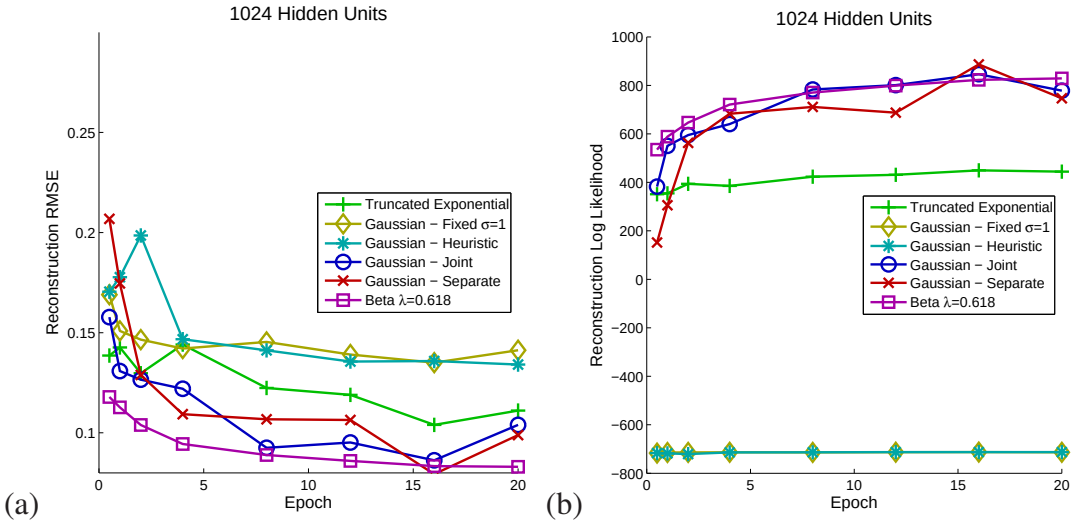

Figure 1: Reconstruction accuracy for different models. (a) RMSE of reconstructed test patches for different stages of training. (b) Log likelihood of reconstructed patches.

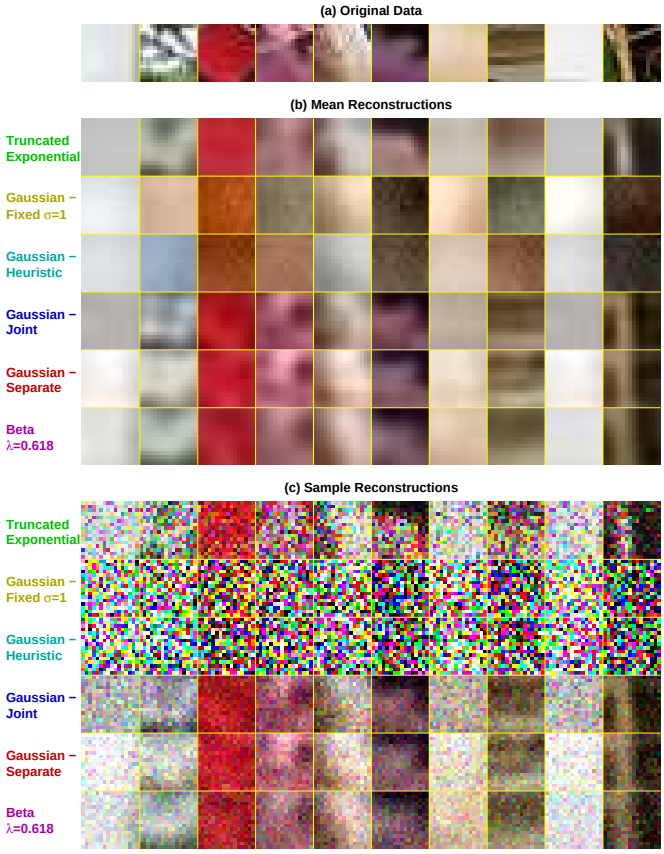

Figure 2: Reconstructions of patches from the test set.

of the hidden states and not the full conditional distribution. Finally, we compute the pixel-wise squared error between the reconstruction and the original patch .

RMSE reconstruction accuracies for the different models (with 1024 hidden units) are shown in Figure 1(a) where the accuracies have been averaged across all test patches. Note that because the RMSE measure uses only the mean of $P(\mathbf{v}|\mathbf{h}^\star)$, the accuracy of the variance of $P(\mathbf{v}|\mathbf{h}^\star)$ is not assessed in these plots. A selection of test patches and their mean reconstructions are shown in Figure 2(a,b).

The truncated exponential does a reasonable job of reconstructing the patches but it is exceeded in performance by all three of the learned-variance models. This leads to the counter-intuitive result that models designed to capture data variance prove to be significantly better at representing the mean. An explanation is that these models learn where they are able to represent the data accurately (e.g. in untextured regions) and where they cannot (e.g. near edges) and hence are able to focus their modeling power on the former rather than the latter, leading to an overall improvement in RMSE. The overall best performer is the Beta RBM which not only has the best average RMSE but also shows much greater stability during training in comparison to the Gaussian models (as may be seen in Figure 1a).

2.7.3 Reconstruction log-likelihood

This experiment is a proxy to the true log-probability of the data. To obtain the true probability of a test patch, one could start a Markov chain from this same patch, run for an infinite amount of time, and compute the log-probability of that patch under the final distribution (the choice of starting point would actually have no influence). Since this would be too expensive, we only consider a unbiased sample of the distribution obtained after one Markov step. We therefore perform the following experiment:

1. given a test patch $\mathbf{v}_{\text{test}}$, we sample a configuration of the hidden states $\mathbf{h}^\star$ from the conditional distribution $P(\mathbf{h}|\mathbf{v}_{\text{test}})$, and then

2. given this configuration of the hidden states, we compute the **conditional probability** of the test patch $P(\mathbf{v}_{\text{test}}|\mathbf{h}^\star)$, which is easily done given the factoriality of this distribution.

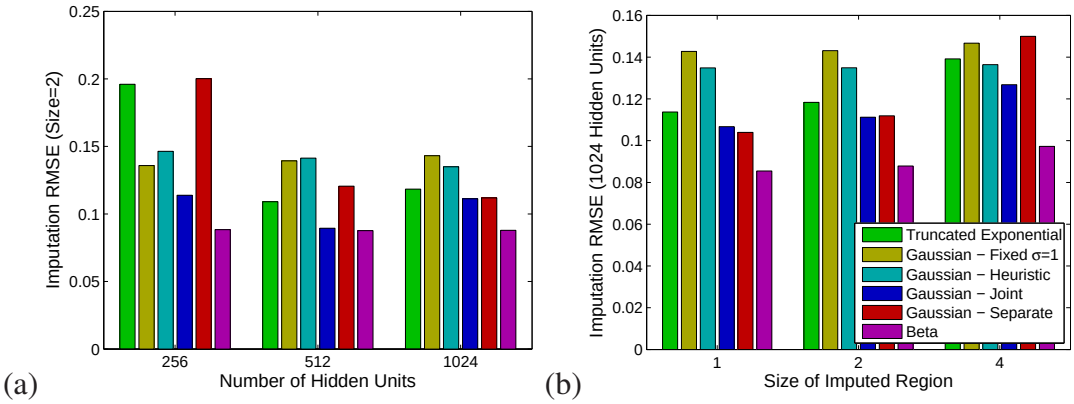

Figure 3: Imputation RMSE for different models, as a function of (a) the number of hidden units and (b) the size of the imputed region.

Results for all models are given in Figure 1(b), again with 1024 hidden units. Unlike the RMSE reconstruction, the log-likelihood jointly assesses the accuracy of the mean *and* variance of the model. Hence, differences from the RMSE reconstruction results indicate models where the variance is modelled more or less accurately. Unsurprisingly, the fixed variance models do very poorly on this metric since they have fixed, large variances. More interestingly, the joint Gaussian model now achieves very similar performance to the Beta indicating that it is be modelling the variance better than the Beta (considering that it modelled the mean slightly worse). This may be due to the Gaussian being light-tailed in comparison to the Beta and hence able to put greater probability mass near the mean.

2.7.4 Imputation accuracy

As a further investigation of the models' abilities to represent the distribution over image patches, we assessed their performance at filling in missing pixels in test patches, a process known as *imputation*. The experimental process was:

1. given a test patch, randomly select a region of 1×1, 2×2 or 4×4 pixels and consider these pixels to be missing,

2. initialize the missing pixels to the mean of the observed pixels, and

3. perform 16 bottom-up and top-down passes to impute the values of the missing pixels. In each top-down pass, the values of the observed pixels are fixed whilst

the values of the missing pixels are sampled from $P(\mathbf{v}|\mathbf{h})$. The number of passes is chosen big enough to allow mixing to occur (bear in mind that we are sampling from the conditional distribution of the unobserved pixels given the observed pixels, which is highly concentrated).

The RMSE between the imputed and true pixel values for the different models are shown in Figure 3(a) for models with differing numbers of hidden units and in Figure 3(b) for different sized imputation regions. Again, the Beta RBM leads to the best performance in all cases with the less stable joint Gaussian RBM typically coming in second.

2.8 Conclusion

Across experiments, the Beta RBM proved more robust and slightly more accurate than all the other types of RBM. We therefore decided to use it to model appearances. Nevertheless, one should bear in mind that there is room for improvement and other, higher quality continuous-valued RBMs may exist.

3 The Masked Restricted Boltzmann Machine

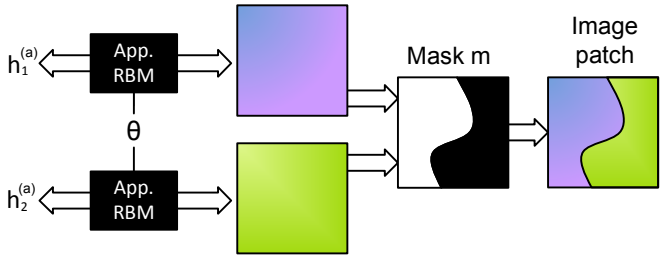

Figure 4: **The Masked RBM.** A masked RBM models an image patch as the composition of two or more latent patches, each generated from a separate appearance RBM with shared parameters θ. The composition is controlled by a mask $\mathbf{m}$, indicating which of the latent image patches is to be used to model each visible image pixel.

An RBM will capture high order interactions between visible units, to the limit of its representational power determined by the number of hidden units. If there are

not enough hidden units to perfectly model the training distribution, one can observe a "blurring" effect: when two input variables are almost always similar to each other and sometimes radically different, the RBM will not capture this rare difference and will assign a mean value to both variables. When modeling the appearance of image patches, any two nearby pixels will exhibit this property (being different only when an edge is present between these two pixels), thus resulting in a poor generative model of image patches (as shown in the $K = 1$ case of Figure 6). To avoid this effect, a standard RBM would require a number of hidden units equal to the product of the number of possible locations for an edge and the number of possible appearances. Not only would that number be prohibitive, it would also be highly inefficient since the vast majority of hidden units would remain unused most of the time. Another, more efficient, way to bypass this constraint of consistency within the dataset is to have K appearance RBMs, each generating a latent image patch $\widehat{\mathbf{v}}_k$, competing to explain each pixel in the patch. Whenever an edge is present, one RBM can explain the pixels on one side of the edge while another RBM will explain pixels on the other side. We say that such a model has K **layers**. To determine which appearance RBM explains each pixel, we introduce a **mask** with one mask variable per pixel (m_i) which can take as many values as there are competing RBMs. The overall masked RBM is shown in Figure 4 and its associated factor graph is shown in Figure 5.

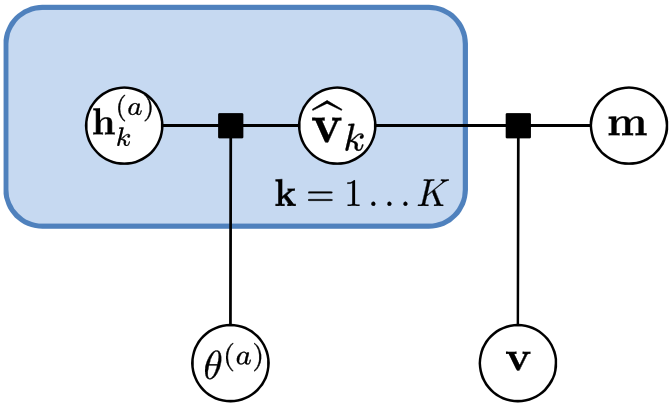

Figure 5: **Factor graph of the masked RBM with a uniform mask prior.** The parameters $\theta^{(a)}$ are outside the plate and thus the same for all RBMs.

To simplify the notation, we shall use a generic form of an RBM (omitting the biases

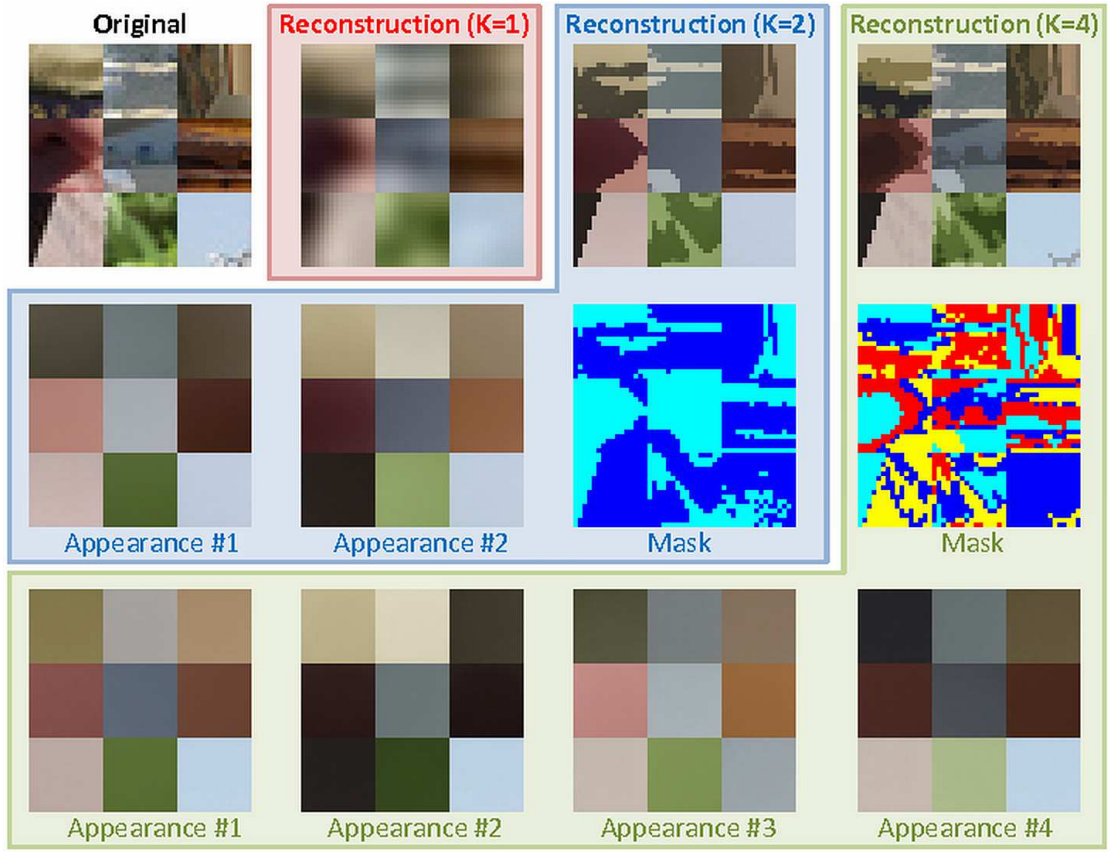

Figure 6: **Reconstructions of nine patches using a masked RBM with $K = 1, 2$ or 4 appearance models.** When $K = 1$, the model is an ordinary Beta RBM and is unable to capture sharp edges in the image. When $K = 2$ or $K = 4$ Beta RBMs are used in a masked RBM, the reconstruction accuracy is much greater and the masks capture the shape of the object in the image. The inferred masks and mean patch from each of the Beta RBMs are shown. The experiment is detailed at the end of section 3. **All models have the same total number of hidden variables.**

for clarity)[4]:

$$\log P(\mathbf{v}, \mathbf{h}) = \sum_{ij} f(\theta_{ij}, v_i, h_j) - \log Z \tag{10}$$

where f depends on the type of RBM chosen (in a binary RBM, we would have $f(\theta_{ij}, v_i, h_j) = -\theta_{ij} v_i h_j$).

In the remaining, we shall also use the following notation:

[4] to include the biases, one would add two functions g (with parameters b_i and v_i) and h (with parameters c_j and h_j)

- since most of the equations will involve all the layers, we will define a shortcut notation: for any variable t defined for each layer k, the set of variables $\{t_1, \ldots, t_K\}$ shall be replaced by $t_{1..K}$
- $\mathbf{v}$ is the image patch
- $\widehat{\mathbf{v}}_k$ is the k-th latent patch
- $\mathbf{h}_k^{(a)}$ the hidden state of the k-th layer. The (a) superscript stands for "appearance" as we will introduce shape layers later on.

Using these notations, given a mask $\mathbf{m}$, the log-probability of a joint state $\mathbf{s} = \{\mathbf{v}, \widehat{\mathbf{v}}_1, \ldots, \widehat{\mathbf{v}}_K, \mathbf{h}_1^{(a)}, \ldots, \mathbf{h}_K^{(a)}\} = \{\mathbf{v}, \widehat{\mathbf{v}}_{1..K}, \mathbf{h}_{1..K}^{(a)}\}$ is equal to

$$\sum_i \left[\log \delta\left(\widehat{v}_{m_i,i} = v_i\right) + \sum_j \sum_k f\left(\theta_{ij}^{(a)}, \widehat{v}_{k,i}, h_{k,j}^{(a)}\right) \right] - \log Z \quad . \tag{11}$$

The first term allows our model to assign infinite energy (and therefore zero probability) to configurations violating the constraint that, if layer k is selected to explain pixel i (i.e. $m_i = k$), then we must have $\widehat{v}_{k,i} = v_i$. To demonstrate the efficiency of using several masks, we infer the mask and hidden states of models with various K given an image, and then "reconstruct" the image using the mask and these hidden states. The inference procedure is described in section A.1 in the appendix. For a fair comparison, we used the same total number of hidden variables for each value of K (accounting for the bits required to store the mask and the hidden units for each appearance model). The reconstruction with $K = 4$ thus used RBMs with many fewer hidden units ($n = 128$) than the one with $K = 1$ ($n = 1024$). From the results shown in Figure 6, we see that it is advantageous to assign a large number of bits to the mask rather than to the appearance. A more thorough evaluation of the masked RBM is presented in section 5.

4 Modeling shape and occlusion

The energy of Eq. 11 can be used to define a conditional distribution given the mask. To get a full probability distribution over the joint variables, we must also define a distribution over the mask. In this paper, we shall consider three different mask models: a uniform distribution over all possible masks, a multinomial RBM which we denote

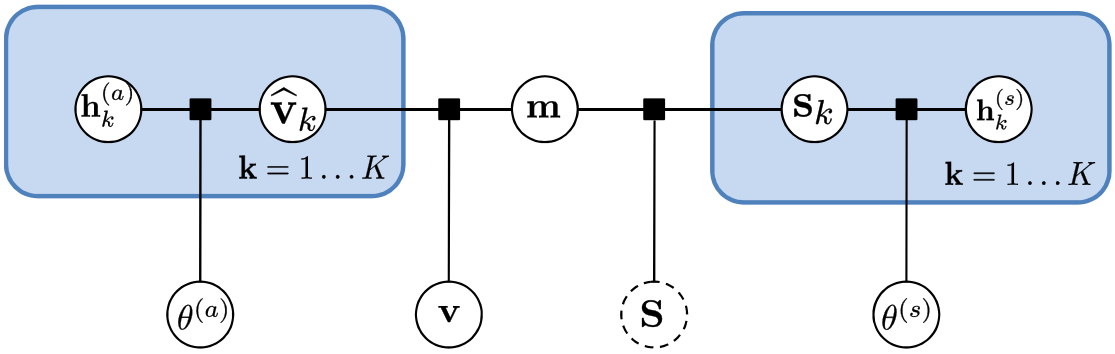

Figure 7: **Factor graph of the masked RBM with a non uniform mask prior.** The ordering S is only used in the occlusion model.

the "softmax" model and a model which has been designed to handle occlusions, which we call the "occlusion-based" model. The latter two models will allow us to learn a model of the shapes present in natural images.

The learning and inference procedures in these models may be found in section A of the appendix.

4.1 The uniform model

The simplest mask model is the uniform distribution over $\mathbf{m}$. In this model, no mask is preferred a priori and the inferred masks are solely determined by the image. We use this model as a baseline.

4.2 The softmax model

The softmax model consists of K binary RBMs with shared parameters competing to explain each mask pixel. Each RBM defines a joint distribution over its visible state $\mathbf{s}_k$, which is a binary shape, and its binary hidden state $\mathbf{h}_k^{(s)}$ (the (s) superscript stand for "shape"). The K binary shapes $\mathbf{s}_k$ are then combined to form the mask m, which is a K-valued vector of the same size as the $\mathbf{s}_k$'s. To determine the value of m_i given the K sets of hidden states $\mathbf{h}_k^{(s)}$ requires computing a softmax over the K different inputs.

The joint probability distribution of this model is:

$$\log \quad P(\mathbf{m}, \mathbf{s}_{1..K}, \mathbf{h}^{(s)}_{1..K}, S) = \sum_k \left[\sum_{ij} s_{k,i} W^{(s)}_{ij} \mathbf{h}^{(s)}_{k,j} + \sum_j c^{(s)}_j \mathbf{h}^{(s)}_{k,j} \right]$$
$$+ \quad \sum_i \left[\log \delta(s_{m_i,i} = 1) + \sum_{k \neq i} \log \delta(s_{k,i} = 0) \right] - \log Z \tag{12}$$

The third and fourth terms state that only one shape may be "on" at any given pixel, and that the index of the selected shape is the value of the mask at that pixel.

Inference is relatively straightforward in this model, but at the cost of poor handling of occlusion. Indeed, this models makes the implicit assumption that all the objects are at the same depth. This gives rise to two problems:

1. when object A is occluding object B, the shape of object B is considered as absent in the occluded region rather than unobserved. As a consequence, the model is forced to learn the shape of the visible regions of occluded layers. For example, with a digit against a background, the model is required to learn the shape of the visible region of the background, in other words, the inverted digit shape.

2. there is no direct correspondence between the hidden states of any single layer and the corresponding object shape, since the observed shape will jointly depend on the K inputs.

4.3 The occlusion model

An occlusion occurs when an object is at least partially hidden by another one. In the occlusion model, we explicitly represent this hiding by introducing an ordering S of the layers ($S(1)$ being the index of the foremost layer and $S(K)$ being the index of the backmost layer), where each layer contains a shape. For this shape to be visible, there must not be any other shape at the same location in the layers above. The joint probability distribution for this model is:

$$\log \quad P(\mathbf{m}, \mathbf{s}_{1..K}, \mathbf{h}^{(s)}_{1..K}, S) = \log P(S) + \sum_k \left[\sum_{ij} s_{k,i} W^{(s)}_{ij} \mathbf{h}^{(s)}_{k,j} + \sum_j c^{(s)}_j \mathbf{h}^{(s)}_{k,j} \right]$$
$$+ \quad \sum_i \left[\log \delta\left(\mathbf{s}_{m_i,i} = 1\right) + \sum_{k:S(k)<S(m_i)} \log \delta\left(\mathbf{s}_{k,i} = 0\right) \right] - \log Z \tag{13}$$

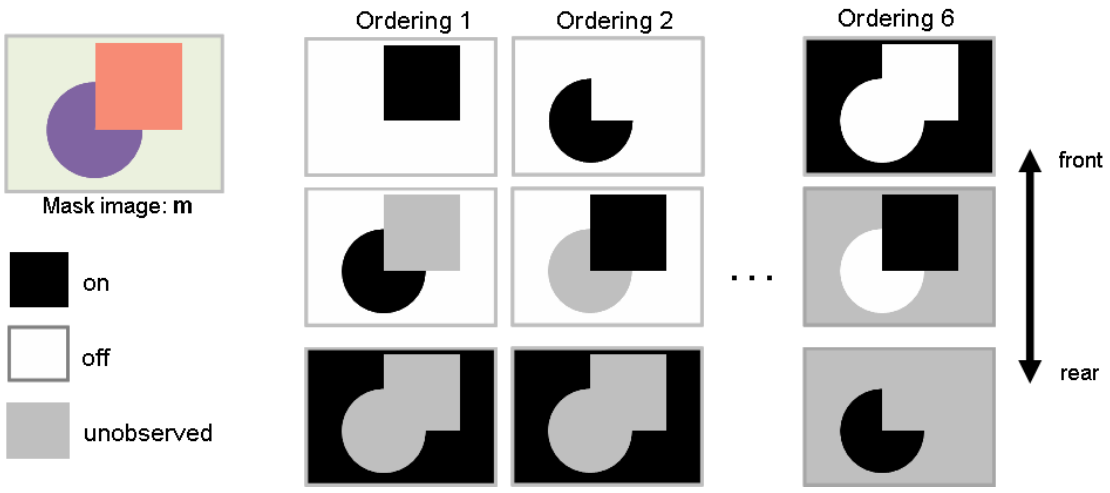

Figure 8: **Depth inference in the occlusion model.** The mask image (top left) comprises three regions, so there are $3! = 6$ possible depth orderings. Together with the mask, the ordering defines which shape pixels $\mathbf{s}_{k,i}$ are observed and which are unobserved. This is illustrated for three of the six possible orderings (white regions: shape off; black regions: shape on; gray regions: shape unobserved). Unobserved pixels (corresponding to $U_{S,k}(\mathbf{m})$ in eq. 19, see appendix) can be "filled-in" by the shape model. Thus, for a shape model that favors circles, squares, and homogeneous backgrounds ordering 1 is preferable to all other orderings (including 2 and 6).

There are two differences to the softmax model:

1. we now have a prior $P(S)$ over the depth ordering (which is chosen to be uniform)

2. if $m_i = k$, then we must have $\mathbf{s}_{k,i} = 1$ (as in the softmax model), but we only require that $\mathbf{s}_{k',i} = 0$ for the layers k' in front of the layer k (rather than for all the layers as is the case in the softmax model). $\mathbf{s}_{k'',i}$ for k'' behind layer k are unobserved (occluded). This idea is illustrated in Figure 8.

With this model, there is a direct correspondence between the hidden states and the shape of the object (see Figure 9).

The description of the inference procedure for the depth ordering S is described in section B.1 of the appendix.

5 Inferring appearance and shape of objects in images

We recall that our goal is to learn a good generative model of images by extracting a factorial latent representation (appearance and shape) of objects in natural images. To assess how well this goal is achieved, we shall try to answer a set of questions:

- how visually similar are the samples from our model to samples coming from the same distribution as the training set? Though poor samples characterize a bad generative model, the converse is not true as samples too close to the training data show a lack of generalization of our model, which is not desirable. Despite the flaws of this "measure", we think it can provide meaningful insight on what has actually been learnt.

- do samples from our model exhibit the same statistics than those computed on test patches?

- are test patches likely under our model?

- did we really factor appearance and shape? Are the latent representations we extract meaningful? Are they independent of the depth ordering of the objects in the image? Are the depth orderings correct?

The first three questions relating samples from our model and test data can be answered both on a toy dataset and a real dataset of natural images. However, a toy dataset offers the additional advantage of providing the ground truth objects from which the patches have been created, which makes it easier to assess the quality of the generative model.

The last questions are trickier to answer in the context of natural images since we have no control over the ordering of the objects. However, there are some natural patches for which there is little ambiguity over that ordering. If the model is able to infer a plausible answer in these cases, this should be a good indicator of the quality of its inference of the depth ordering of the objects (and thus a measure of the invariance of the inferred latent shapes to this ordering).

5.1 Training

This section describes the training procedure for the masked RBM, as this model proved much more complicated to train than a standard RBM. Details on the datasets used are provided in the next sections.

The training was done in several stages of increasing complexity for efficiency reasons:

1. we first trained a single, unmasked, RBM until low-frequency filters appeared. This allowed us to quickly obtain a good initialization for the filters typically obtained in the masked RBMs (since the edges are captured by the masks, none of them are high-frequency) by avoiding having to infer the mask at each iteration

2. initializing with the filters from the previous step, we then trained a masked RBM with a uniform mask model (which means we only trained the appearance RBM) and $K = 2$. Using a lower K allows us to speed up inference while still providing good initial filters for the final stage. K was then switched to 4 until parameters converged. The reason why we trained the appearance model in the context of a masked RBM is to avoid wasting capacity modeling complicated shapes which will be handled by the mask.

3. we froze the parameters of the appearance RBM and we learned a softmax shape model (whether the final model is the softmax one or the occlusion based one) using as training data the masks inferred at stage 3. We suspect the reason why we cannot use the occlusion based model at that point is that the inference of the depth ordering is poor when the shape model is not close to convergence.

4. we fine-tuned both the appearance RBM and the shape RBM by performing the joint inference of the parameters of both models (the masks being inferred at each iteration using the current state of the RBMs), using the correct shape model.

Bootstrapping allowed for faster learning of this complex model. Also, experiments seemed to indicate that it helps finding a better global solution and avoiding undesirable local minima.

5.2 Toy masks dataset

The toy masks dataset is composed of 4000 14×14 mask images generated from the superposition of an MNIST digit (from the class "3") and a shape (a circle, a square or a triangle). In this dataset, neither digits nor shapes are shown in isolation, and each digit example appears only in exactly one image. Since the digit is in the background on half of the patches, half of the digit examples are only partially visible. Samples of this dataset (which are masks) are shown in Figure 9a: each pixel can take three values (represented by the colors red, green and blue), one for each object in the patch (the background being the third object). Which color is assigned to each object is irrelevant (the actual values are not used to infer the depth ordering), it only matters that they are assigned different colors.

5.2.1 Quality of the generative shape model

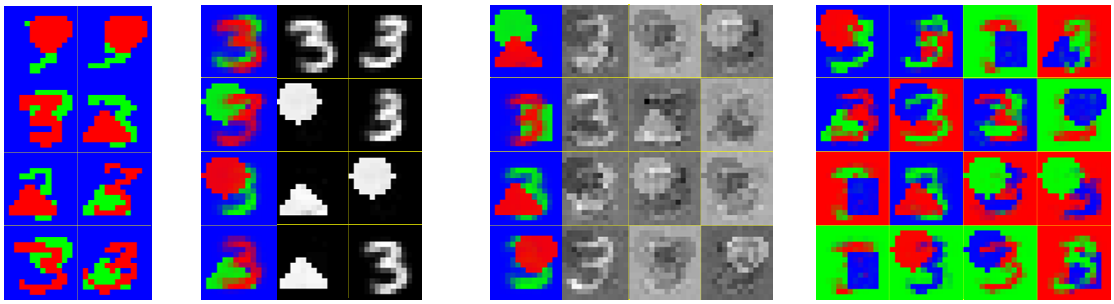

Figure 9: **Learning shapes under occlusion**: (a) Training data. (b1) Samples from the occlusion model (20 hidden units per layer), obtained by composing latent shapes (b2 and b3). (c1) Samples from the softmax model with 70 hidden units (left-most column) using contributions of the three layers (c2, c3 and c4). (d) Samples from the softmax model with 20 hidden units per layer. The softmax model can compensate for its limitations by using more hidden units, but its performance quickly decreases when it has limited capacity, yielding invalid samples (in the bottom right sample, the "3" goes through the square).

We trained our mask model using three layers ($K = 3$). Figure 9 shows samples from the occlusion model with 20 hidden units (b), the softmax model with 70 hidden

units (c) and the softmax model with 20 hidden units (d). Samples from the occlusion model are drawn by sampling from the two RBMs governing the top-most and second-most layer independently and then composing these samples as prescribed by eq. 13. One can see that, when using 20 hidden units, the samples drawn from the occlusion based mask model are much more convincing than those drawn from the softmax model. Indeed, the latter generated samples with improper occlusions or deformed digits. It is also interesting to note that the occlusion model generalized to samples not seen in the training set, like the two MNIST digits occluding each other. Furthermore, columns (b2) and (b3) show samples of the latent shapes, proving that the occlusion model learnt a model of the individual shapes — despite the fact that it has never seen them in isolation.

In the softmax model, on the other hand, the layers cooperate to generate a particular image of occluding shapes. It is not possible to sample from the individual layers separately, but one can still inspect the inputs to the three layers of visible units which are tied together by the softmax. These inputs are shown in Figure 9 (c2, c3 and c4). It is clear that no shape is generated by a single layer but that all three layers have to interact. In the first row, for instance, all three inputs contain a "3" (either with positive or negative weights) despite it being absent from the resulting sample. Though harmful (because they require additional modeling power), these cancelations are inevitable in the softmax model. While the occlusion model learns about the individual image elements, the softmax model has to represent all their possible arrangements explicitly, which is less efficient and thus requires a larger number of hidden units. This also leads to a set of hidden units which is far less indicative of the shape in the image than in the occlusion model.

5.2.2 Sensitivity to occlusion

To assess the importance of the difference in representation between the softmax and the occlusion mask models, we created pairs of images containing one digit and one shape (the same digit and the same shape were used in both images of a pair). In the first image, the digit was in front of the shape and in the second image, the shape was in front of the digit. We compared the inferred shape latent variables for the two cases and computed their root mean squared difference. As our main motivation is to recognize

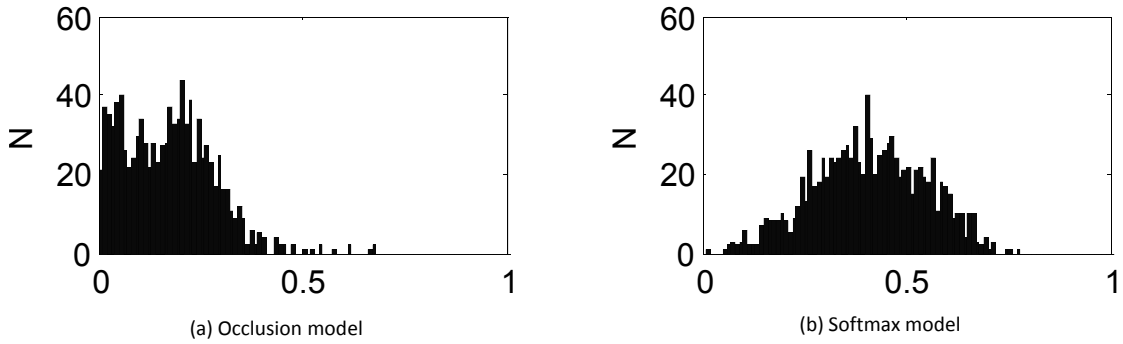

Figure 10: Histograms of the root mean squared differences between the activation of the hidden units inferred depending on the relative positions of the MNIST digit and the shape in the test image for the softmax model and the occlusion model.

objects whether or not they are occluded, we would like the shape latent variables to be as similar as possible in the two cases. Unsurprisingly, the occlusion based mask model clearly outperforms the softmax model, as may be seen in Figure 10. Furthermore, in our experiments, the occlusion model inferred the correct ordering more than 95% of the time (chance being 17%, as there are three layers and six possible orderings).

This toy dataset emphasizes the need for modeling occlusion when extracting a meaningful representation of the shapes present in images.

5.3 Natural image patches

The experiments on toy data demonstrated that the occlusion model is able to learn and recognize shapes under occlusion and is able to perform depth inference given a mask image with occluding shapes. The second set of experiments on natural images assesses the joint model consisting of the shape and the appearance model. For this purpose we trained the full model with K=3 on 10K 16×16 patches extracted from natural color images. The mask model used in all these experiments is the occlusion model. The appearance RBM had 128 hidden units and the shape RBM had 384 hidden units.

As outlined above, our criteria for assessing the model on this dataset were

1. whether samples from the model looked qualitatively similar to the natural image patches that we had trained the model on (section 5.3.2)

2. whether samples from the model exhibited the same statistics as natural image

patches (section 5.3.3)

3. whether it would learn to use simple shape-based depth cues (section 5.3.4).

5.3.1 Sampling from a confident continuous-valued RBM

When learning the appearances of the objects with the Beta RBM, each expert becomes extremely confident. This is even more striking in the masked context where the noise model does not need to explain the sharp variations of appearance at the boundaries of objects. While this is a good thing from a generative point of view, it leads to a very poor mixing of the Gibbs chain. Indeed, as the conditional distributions $P(\mathbf{v}|\mathbf{h})$ become very peaked, so do the distributions $P(\mathbf{h}|\mathbf{v})$ and the relationship between $\mathbf{v}$ and $\mathbf{h}$ becomes quasi-deterministic. This makes it hard to:

- learn the parameters in the final stage as the samples from the negative chain are highly correlated between consecutive time steps
- draw samples to assess the quality of the generative model
- compute an accurate approximation to the partition function to estimate the log-probability of test patches.

The first issue was dealt with by using tempered transitions (Salakhutdinov, 2009) twice per sweep through the training set. To improve sampling, we trained a binary RBM on top of our Beta RBM. As such RBMs mix much more easily, we could draw samples by running a Gibbs chain in this top binary RBM, before performing a top-down pass in the bottom Beta RBM. Unfortunately, even then, AIS (Salakhutdinov and Murray, 2008) proved unreliable. We therefore decided not to include log-probability results whose validity we could not properly assess.

5.3.2 Visual assessment of the samples

Sampling from the mask model was performed by sampling the binary RBMs in the shape layers and composing them according to a randomly chosen depth ordering. Samples are shown in Figure 11, right. Though they do not exhibit as much structure as true natural image patches (Figure 11, left), the presence of multiple sharp edges makes

them look much more convincing than the typical blurred samples one may obtain from a single RBM. Moreover, the samples clearly capture important characteristics of the training patches (such as the dominance of homogeneous regions and the shape of the boundaries of these regions), despite the relative simplicity of the model and the fact that K was chosen to be small.

Figure 11: True natural patches (left) and samples from the masked RBM (right).

5.3.3 Image statistics

We first assess the quality of the samples from the masked RBM by computing the responses of patches (either natural ones or samples from our model) to a set of Gabor filters, even and odd, at different spatial frequencies and orientations. Before computing the filter responses, we converted all the patches to grayscale. The filters are shown in Figure 12. We compared four kinds of patches:

- natural patches
- patches sampled from the masked RBM. The appearances and the shapes are true samples from the model. This model used $K = 3$ layers
- patches sampled from a single, unmasked, RBM
- patches generated from Gaussian noise with the same covariance as natural patches.

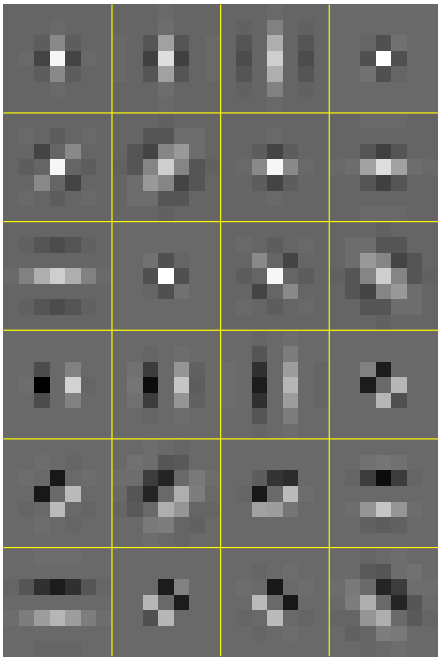

Figure 12: Odd and even Gabor filters used to compute statistics over patches. The layout of these filters corresponds to the one used in Figure 13.

The results (displayed as log-probability of each response value) are shown in Figure 13, where the layout corresponds to the layout of filters in Figure 12. The responses for patches from the masked RBM (in blue) have much heavier tails than those for patches for the unmasked RBM (in red).

The samples were obtained by running a Gibbs chain for 5000 steps for the appearance RBM (both for the masked and the unmasked RBM) and 20000 steps for the shape RBM. Due to the pixel-independent noise model, the peak at 0 is underestimated for these two models (nearby pixels have an extremely low probability of having the same value, unlike true image patches). If we run a Gibbs chain in the masked RBM for the same amount of time but, at the last step, we take the mean activation of the visible units given the binary hidden states (rather than a sample), we get the filter responses shown in figure 14 (only the region near the origin is shown). The tails remain the same but the peak at 0 is more pronounced, closely matching the ones obtained with true image patches.

Again, we wish to emphasize that the model has never been trained to match these statistics. The improved matching, in particular the heavy tails, arose naturally with the

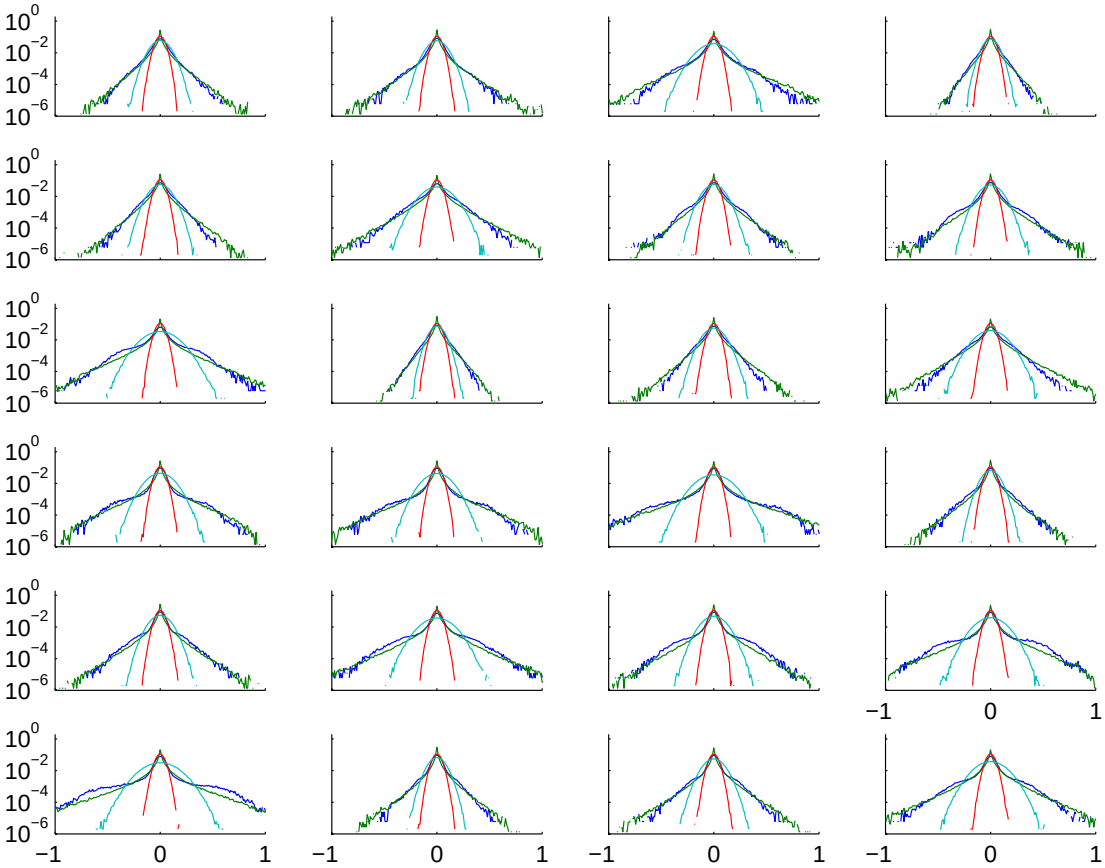

Figure 13: Filter responses for various kinds of patches: **Green**: real image patches. **Blue**: samples from the masked model with $K = 3$. **Red**: samples from the appearance model only (no shape). **Cyan**: Gaussian noise with the same covariance as the real patches. Whereas the samples generated from a single RBM exhibit a Gaussian-like response, the response obtained from samples from the masked RBM closely match those obtained from real image patches.

use of a mask.

5.3.4 Inference of relative depths based on shape

The goal of this experiment is to investigate whether learning an efficient representation of the data leads to the model being able to reason about relative depths. For this purpose we chose a simple scenario shown in Figure 15: patches that contained simple shape-based depth cues were extracted from an image (a). For each patch, the model inferred a segmentation mask with up to K=3 regions (b.1), a relative depth ordering (front to

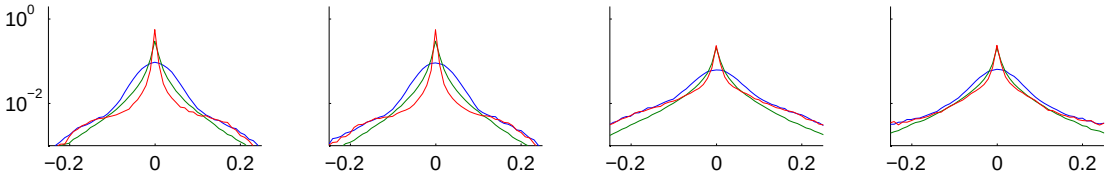

Figure 14: Difference between sampled and mean activations in a zoomed-in region close to the peak, for the first four Gabor filters. **Green**: real image patches. **Blue**: samples from the masked model with $K = 3$ where the activations of the visible units have been sampled given the binary hidden states. **Red**: samples from the masked model with $K = 3$ where the activations of the visible units are the average of the activations given the binary hidden states. Due to the smoothness induced by the averaging, the peak at 0 is much more pronounced and is much closer to the one obtained with real image patches. Similar results were obtained for the other Gabor filters.

back: red — green — blue), the potentially partially unobserved shapes of the two rear-most layers (b.2) and the appearances of the three layers. Knowledge of the latent shapes allows for removing the foreground shape and imputing the missing parts of the second layer shape (c.1 and c.2: segmentation mask with two layers and imputed image respectively). For the examples shown, the model inferred depths orderings and latent shapes largely consistent with the full image. We observed that the patches for which this was not the case often exhibited matting or shading, situations not accounted for by the model. Despite these limitations, it is interesting to observe that learning an efficient representation of the data also appears to have made the model pick up certain simple depth cues, despite never having received depth information with the training data.

6 Field of Masked RBMs

Since our ultimate goal is to build a generative model of natural images, we need to move from the patch level to the image level. This can be achieved by dividing the image into a set of non-overlapping patches, each modeled with a masked RBM with shared parameters. However, this approach leads to artifacts at the patch boundaries, since correlations between pixels either side of these boundaries are ignored. These artifacts appear because the K patch appearance models that each pixel chooses between

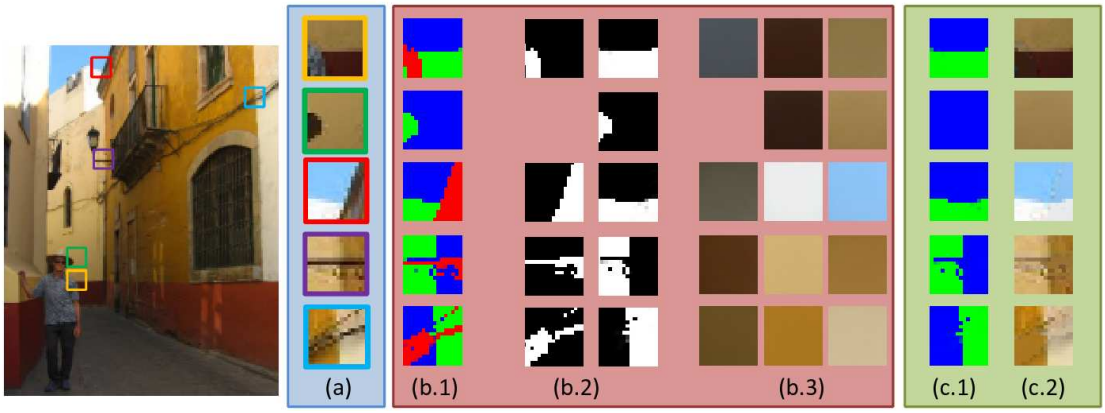

Figure 15: Starting from natural patches (a), we inferred the ordering of the layers (b.1), the latent shapes of the 2 frontmost objects (b.2) and the 3 latent images (b.3). This allows us to recreate a mask image without the foreground (c.1) and the associated patch (c.2).

are aligned, so that their patch boundaries are in the same place. A better solution is obtained by noting that the patch models can be laid out on the image in an arbitrary manner as long as each pixel is covered by at least one patch. In particular the patch models overlapping with (and thus competing for) a particular image pixel do not have to be aligned. Also, the number of patches overlapping with different pixels could vary in principle. In practice it is nevertheless desirable to cover the image with patch models in a regular manner and to cover each pixel with the same number of patches. One way to achieve this is shown in Figure 16. Here, the image is tiled by K grids of abutting patch models. In each grid the patches are non-overlapping and cover all pixels in the image. Across different grids the patch boundaries are spatially offset so that no two patches are fully aligned. In this model, we only get coarse translational invariance (with $K = 4$ and a patch size of 16×16, our model is invariant to translations of 8 pixels or multiples thereof).

Thus the image is covered with overlapping appearance RBMs (and possibly corresponding shape models), arranged such that each pixel is covered by exactly K RBMs. Figure 16 shows a field of masked RBMs, with two of the overlapping appearance RBMs highlighted. The set of mask variables now form a *mask image* with a value for each image pixel indicating which of the K overlapping models it is explained by. It should be noted that this model is a mixture rather than a product of appearance RBMs,

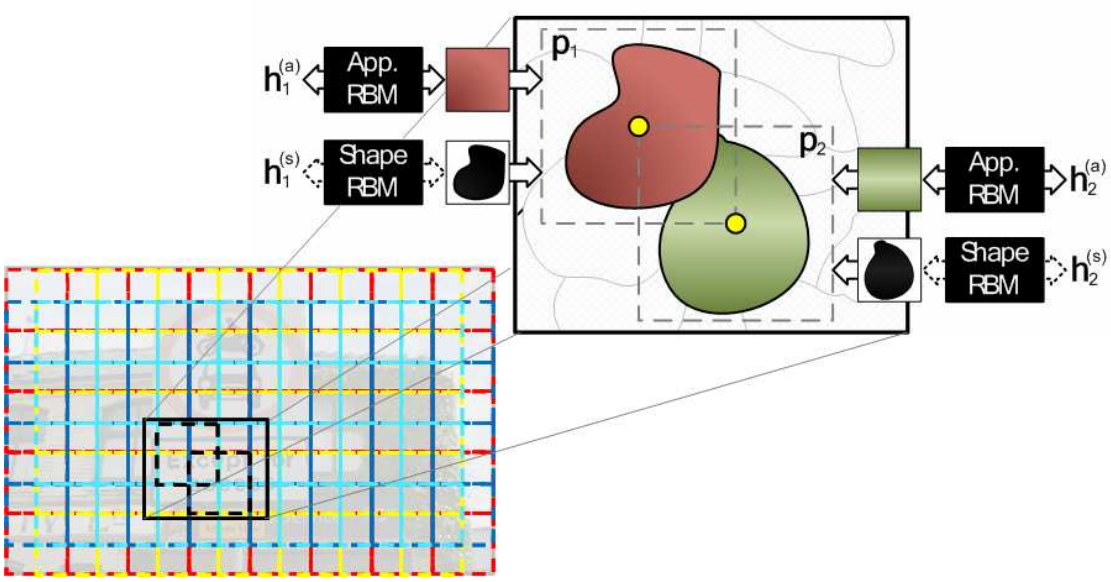

Figure 16: **A field of masked RBMs, where an image is represented using a set of overlapping patch models.** *Left*:The image is covered by K (here $K = 4$) grids of non-overlapping, abutting patches (each grid is shown in a different color: red, yellow, cyan, blue). The different grids are spatially offset so that the patch boundaries in different grids do not align and each pixel is covered by K partially overlapping patches that compete to explain the pixel. *Right*: Blow-up of the interaction between two overlapping patch models. Competition between patch models leads to a segmentation of the image into "superpixels", with one superpixel per patch. The appearance and the shape of each superpixel are modeled by separate RBMs.

in contrast, for instance, to the Field of Experts model of Roth and Black (2005).

Inference is done in the same way as at the patch level with the one difference being that the patch models competing for a particular pixel are no longer aligned. This introduces long-range dependencies between spatially separated patches, so that inference has to be performed on the entire image simultaneously. While this makes perfect sense from a probabilistic point of view (in the general case, one has to take the whole image into account to understand part of it), the result is slower learning and inference.

Figure 17 is the equivalent of Figure 6 for full images. It shows the reconstruction of an image (that is, the image generated using the hidden states inferred from the

Figure 17: **Reconstructions of an image under a field of masked RBMs with differing numbers of appearance layers.** Each appearance layer is represented by a grid of non-overlapping Beta RBMs. In each case the reconstruction uses 4 bits per pixel. The reconstruction quality is highest for $K = 4$ indicating a good trade-off between representing appearance and shape of objects in the image.

original image) using various numbers of layers and a uniform mask model. As for the experiments depicted in Figure 6, we used the same number of hidden variables for each value of K (4 bits per pixel). The RBMs used in the appearance model with $K = 4$ thus have only 128 hidden units whereas those used in the model with $K = 1$ have 1024 hidden units. The patch size is 16×16 pixels for all K.

Both shape models discussed in the previous section (i.e. the softmax- as well as the occlusion-model) can be used at the image level. Figure 18 shows that using such a shape model yields more coherent regions for the mask image without significant loss in reconstruction accuracy. The occlusion model leads to a particularly appealing interpretation at the image level: Each patch model can be thought of as an independent expert modeling shape and appearance of an image patch. It consists of an appearance RBM that determines the color – or more generally texture – of a patch and a binary RBM that determines its shape, as is illustrated in Figure 16. An image is generated by covering it fully with such patches in an occluding manner. In particular – and perhaps

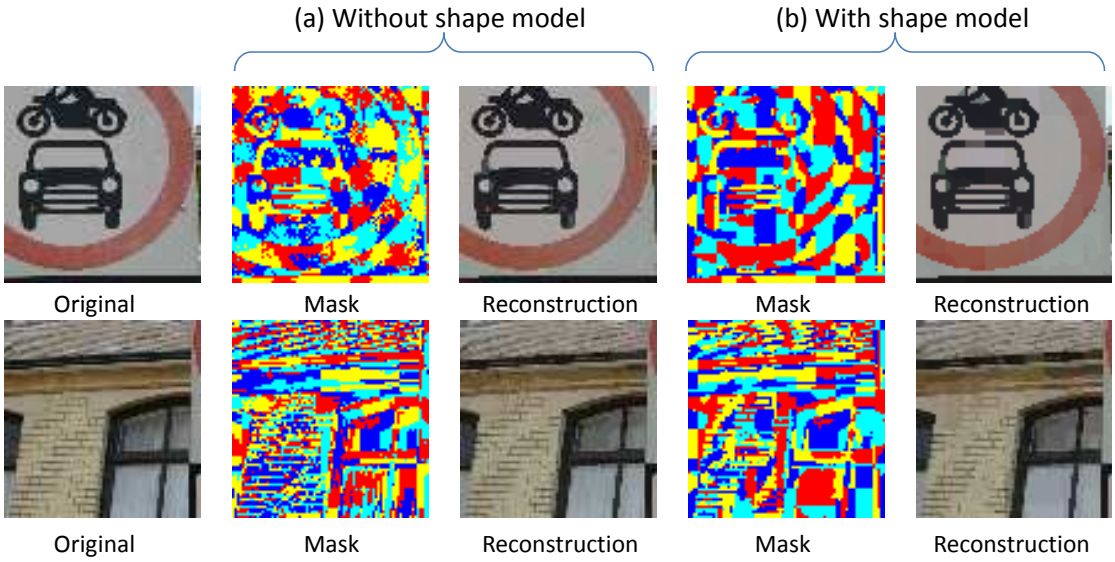

Figure 18: **Field of masked RBMs with and without RBM softmax shape model.** (a) Without shape model, the mask gives the best reconstruction RMSE but is not coherent. (b) With a shape model, the inferred mask is much more coherent, whilst preserving thin structures like brick patterns. Reconstruction quality is very close to the one of (a).

surprisingly – inference with the occlusion model can still be performed efficiently for full images: Even though each image is explained by a potentially large number of patches, each indiviudal patch overlaps only with a small number of neighbors (e.g. for $K = 4$ and the global patch layout shown in Fig. 16, each patch overlaps with 8 neighbors). Thus, instead of determining a global depth order of all patches (which would clearly be infeasible) it is sufficient to infer the depth of each patch relative to its neighbors. The depth of a particular patch given a fixed relative order of its neighbors can be determined following the principles described for image patches in Section 4.3; the full local ordering of all patches covering the image is determined in an iterative manner by considering each patch in turn (see Appendix for details).

6.1 Experiments

Once again, although the reconstruction of test data gives some information about the quality of a generative model, it has severe shortcomings. We shall thus repeat some experiments done at the patch level to show how the main properties of the algorithm

have been preserved despite operating at the image level.

6.1.1 Inferring depths on toy data

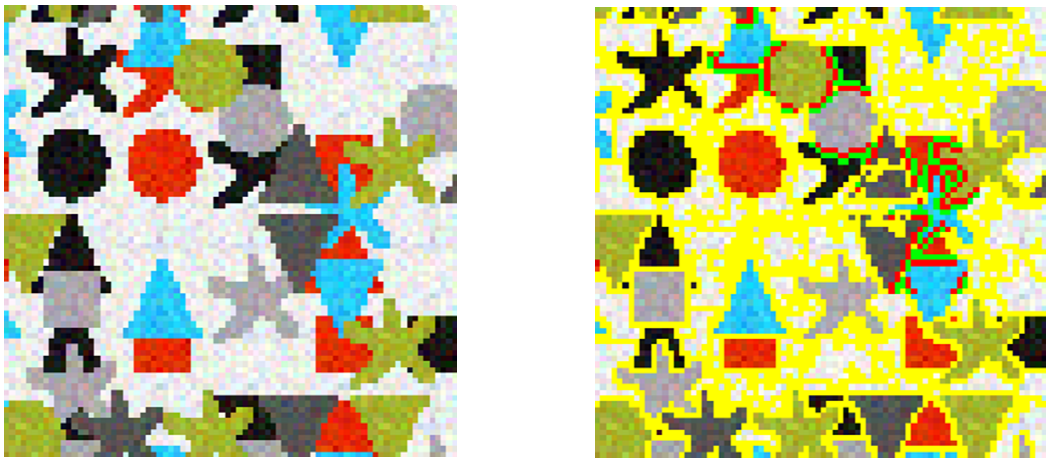

Figure 19: **Left**: training RGB image for the field of masked RBM. There are five different shapes and five different colors. No two overlapping shapes have the same color. Most of the shapes are explained by only one superpixel. **Right**: inferred segmentation and depths. Areas explained by a given superpixel are delimited by yellow lines. In several cases, the inferred relative depth is also displayed: the red line is on the inside of the shape and the green line is on its outside. Therefore, when two shapes overlap (for instance, the green circle and the blue triangle at the top of the image), the red line of the object in the back is cut by the red and green lines of the object in the front (in this case, the blue triangle is in the back). The model inferred the correct depth ordering for all the shapes.

We shall start by assessing the validity of our model on toy data. We focus our attention on three components:

- the allocation of objects to masked RBMs. Namely, are objects fully captured by the RBM they are centered on? Are RBMs explaining only parts of objects?
- how robust is the depth inference between overlapping objects?
- how good are the shape and appearance models learnt using entire images?

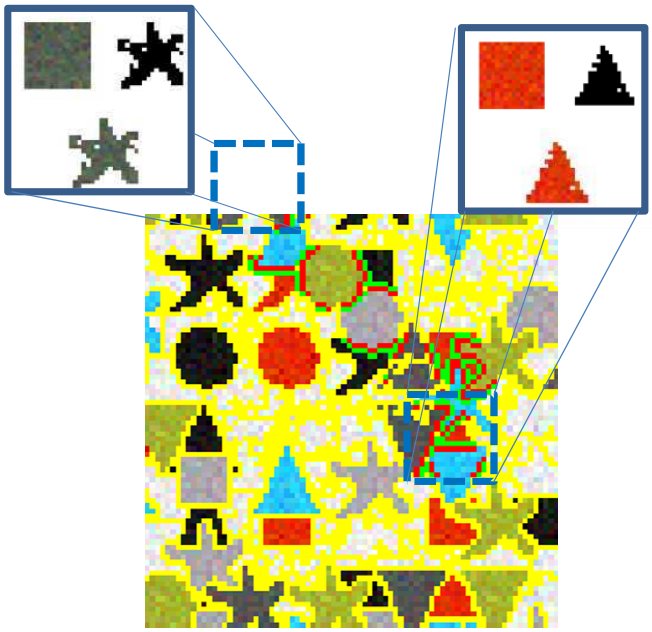

Figure 20: Inference of the shape and appearance of two barely visible objects (one being occluded by another object, the other sitting at the edge of the image). The inferred shapes are very close to the correct answer.

For this purpose, we trained our field of masked RBMs with $K = 4$ on 1000 80×80 images (corresponding to 144 overlapping 16 by 16 patches) composed of five different shapes with varying colors placed randomly in an overlapping fashion against a uniform background (see Figure 19, left for an example; note that shapes were aligned with the patch-grid). We allowed for 20 hidden units for the shape model.

After training, we verified whether the shape model had indeed learned about the shapes comprising the images by sampling from the binary RBM directly. A selection of random samples is shown in Figure 21. Indeed, even though most shapes are only partially visible in the training images (and have varying colors), the shape model has recovered the five templates shapes correctly. Figure 19, right, shows the segmentation inferred with the fully trained model for the image shown on the left. Yellow outlines show the boundaries of objects captured by each masked RBM (patch model). These boundaries indeed reflect the shapes comprising the image (note that the background is segmented in a largely arbitrary manner). Segmentation is obviously not a very difficult task given the image at hand. More interesting is the simultaneously inferred relative

depth of the different image regions and the latent representation inferred for each patch model. The relative depths are shown for a subset of segmentation boundaries which are double-marked with red and green lines. The red side of the boundary points towards the region that has been inferred to be in front; the green side towards the one that is inferred to be in the back. Figure 19 further shows the inferred latent shape and appearance for two of the patch models representing the image (indicated by the blue squares). In both cases the true shapes (a gray star and a red triangle) are barely visible in the image (see also Figure 19). Nevertheless the model correctly infers the appearance and, importantly, completes the partially occluded shape (Figure 20). It is this ability to correctly complete occluded shapes that drives the depth inference.

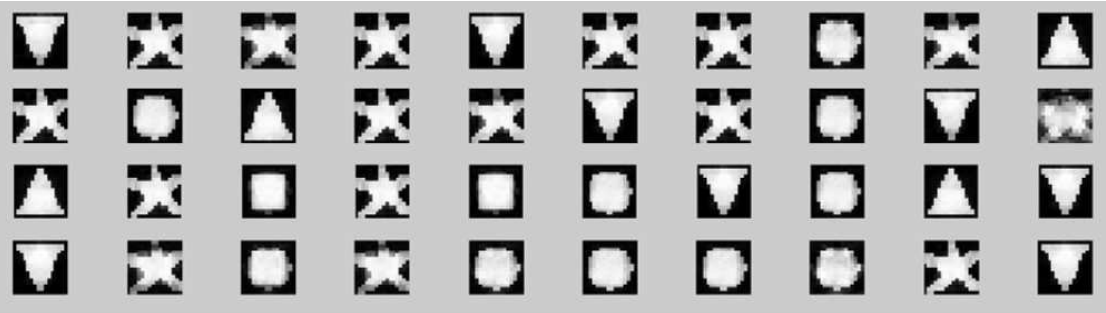

Figure 21: Samples generated from the shape model learnt using the training image from Figure 19, after running a Gibbs sampler for 5000 steps. The images shown are the probabilities of the binary visible units given the binary states of the hidden units. Though most of the shapes are occluded, the samples closely match them. One can see that the model has some difficulties distinguishing the square from the circle.

6.1.2 Image editing

We shall now show how the field of masked RBMs may be used to edit entire images, rather than single patches like in Figure 11. We chose an image where the relative depth of the object in the front could be inferred from local visual cues (bear in mind that, though this is a model of entire images, the depth is only inferred using neighboring patches). As opposed to the experiments on toy data of the previous section, this image was not part of our training set.

Figure 22 shows the original image, as well as the inferred depth of each element

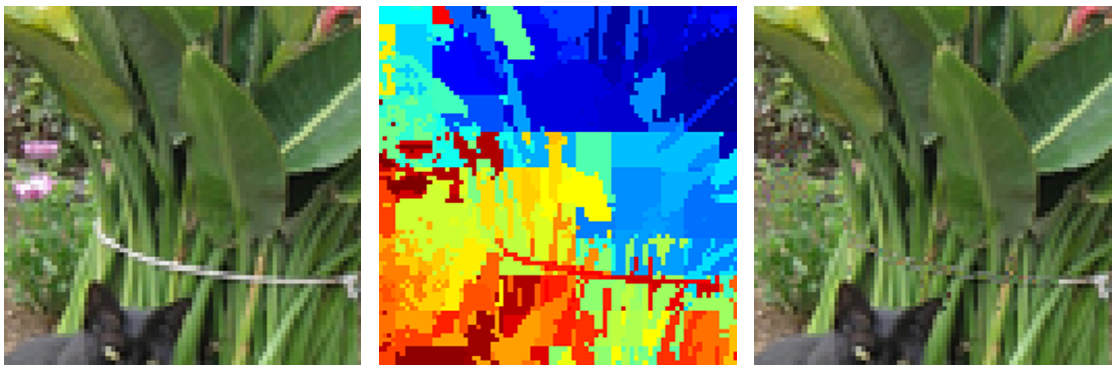

Figure 22: **Left**: training image for the field of masked RBMs. **Middle**: inferred segmentation and depths. Colors denote the relative depth of the region assigned to a given superpixel. Blue regions are in the back (relative to the neighboring superpixels) and red regions in the front (relative to the neighboring superpixels). Most of the shapes are explained by only one superpixel and the relative depths of overlapping shapes have all been correctly inferred. **Right**: edited image after having removed the string around the bunch of flowers (whose extent is determined by the inferred segmentation) and sampled the missing pixels according to their conditional distribution given the observed ones.

(red being in front and blue being in the back). Again, one should focus on the relative depths of nearby objects in this inference. One can notice that the string attaching the bunch of flowers is clearly identified as being the frontmost object. Though identified as being in the front, the rightmost part of the string has been wrongfully separated from the rest of the string. This is due to the shadow on the string which makes it virtually indistinguishable from the background (when one uses only local visual cues).

From there, one can decide to remove this object. In doing so, we get an image where all the pixels are observed, except for those which were underneath this frontmost object (see Figure 22, left). As done previously, we may now sample these pixels from the conditional distributions of the RBM involved in these patches. The resulting image may be seen in Figure 22, on the right.

Except for a few pixels, the resulting image looks convincing and the stems have correctly been inferred.

Interpretation as a superpixel algorithm

The masked field of RBMs learns to represent an image as a number of regions each of which can be explained by a single appearance RBM. These regions can be thought of as *superpixels* although they differ from previous kinds of superpixel in that they are not required to be contiguous but merely constrained to lie within the boundary of a patch. Also, they have high-order shape priors that have the potential to capture complex shapes, such as digits or letters. For example, in Figure 17, for $K = 2$ the same superpixel is used for the white background both inside and outside the 'p' of 'except'. Such non-contiguity makes particular sense when dealing with occlusion, since the same superpixel can be used to represent part of an object either side of a narrow occlusion.

7 Conclusion

The contributions of this paper are as follows. First, we provided an empirical comparison of a range of RBMs able to model continuous data, showing that properly modeling the variance dramatically improves the quality of the model. We then introduced the masked RBM, a generative model that works with the assumption that natural image patches are composed of objects occluding each other. In this model, each object is factored into an appearance and a shape, over which we made no prior assumptions. This proved to be a much more accurate model of image patches than the standard RBM, while still allowing for efficient inference. We demonstrated how it was able to infer the depth of objects in natural scenes using only learnt visual cues. We also showed that properly dealing with occlusion was essential for a good latent representation of objects. Last, composing the masked RBMs into a field, we were able to extend our model to large images while retaining the properties observed at the patch level.

We believe the abilities to deal with occlusion, to model generic shapes and appearances, and the applicability to large images are central to a generative model suitable for a broad range of images. Inspired by previous works which dealt with a subset of these properties, we provided a unified, comprehensive probabilistic framework which, while powerful, remains computationally tractable (though still expensive). We hope that this will encourage the community to build richer, more powerful models, with the

ultimate goal of approaching the capacity of the human visual system.

8 Future work: the Deep Segmentation Network

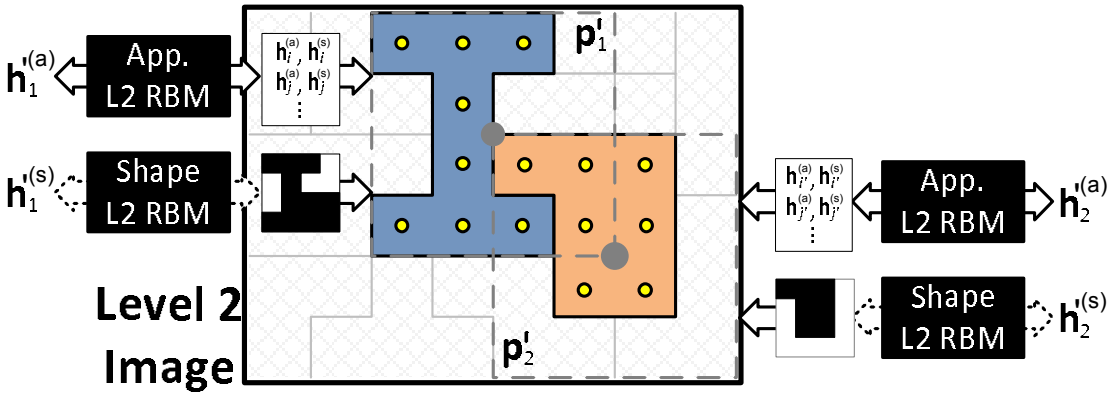

Figure 23: **The second level of a DSN.** The second level of a DSN is a field of masked RBMs of the same structure as the first level (Figure 16) but where the input image 'pixels' are the feature vectors of the first level superpixels $(\mathbf{h}^{(a)}, \mathbf{h}^{(s)})$.

We have shown how a field of masked RBMs is able to decompose an image into superpixels and model the shape and appearance of each superpixel using separate sets of hidden variables, even under occlusion. The next stage of this research is to learn how these superpixels fit together into object parts and how object parts go together to form objects. To do this, we can follow the approach of Deep Belief Nets and combine multiple fields of masked RBMs in a hierarchical model, which we call a Deep Segmentation Network (DSN). The idea is to treat the superpixels learned by the first field of masked RBMs as input "pixels" for a higher-level field of masked RBMs. For example, the superpixels learned in the previous section are associated with patches laid out on a regular 8×8 grid. Hence, we can construct a new "image" one eighth of the size of the original image where the "pixels" are 512 bit feature vectors (384 shape + 128 appearance) rather than RGB values. We can train a second-level field of masked RBMs on a set of such images, where the appearance models are now binary RBMs, as shown in Figure 23. The overlapping patches of the second level cover multiple first level superpixels and hence learn how the shape and appearance of nearby superpixels go together. Mask images will also be inferred for the second level, leading to second

level superpixels which merge a number of first level superpixels. This process can be repeated by adding additional levels to the DSN until the entire image belongs to a single superpixel. This formulation gives rise to a tree-structured hierarchy in which each lower-level node (pixel) is connected to exactly one node in the next level. This hierarchy is, however, not fixed: through the mask which determines to which superpixel pixels are associated, DSNs define an image-dependent parse tree of the input image, similar to Dynamic Trees (Williams and Adams, 1999; Storkey and Williams, 2003). However, DSNs are able to define richer and more complex priors over such parse trees than was possible with DTs.

Preliminary results show that using deeper DSNs leads to meaningful higher level superpixels whilst increasing accuracy on a segmentation task. We believe this is due to the capacity of the higher layers to capture longer range dependencies, allowing parts, entire objects and object context to be captured.

Deeper DSNs will require very large image training sets in order to learn about the range and variability of objects in natural images. Large scale training of deep DSNs is a significant research and engineering challenge that will require extensive parallelisation, in combination with novel methods for learning from vast image data sets. In future we will pursue this goal, with the aim of learning generative models that start to capture the daunting complexity of natural images.

Acknowledgments

The authors would like to thank Chris Williams for his support and help and Iain Murray for insightful comments.

References

Ackley, D. H., Hinton, G. E., and Sejnowski, T. J. (1985). A learning algorithm for Boltzmann machines. *Cognitive Science*, **9**, 147–169.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, **39**(1), 1–38.

Freund, Y. and Haussler, D. (1994). Unsupervised learning of distributions on binary vectors using two layer networks. Technical Report UCSC-CRL-94-25, University of California, Santa Cruz.

Frey, B. J. and Jojic, N. (2003). Learning appearance and transparency manifolds of occluding objects in layers. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society Press, Los Alamitos, CA.

Guo, C.-E., Zhu, S.-C., and Wu, Y. N. (2003). Modeling visual patterns by integrating descriptive and generative methods. *Int. J. Comput. Vision*, **53**(1), 5–29.

Guo, C.-E., Zhu, S.-C., and Wu, Y. N. (2007). Primal sketch: Integrating structure and texture. *Comput. Vis. Image Underst.*, **106**(1), 5–19.

Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural Computation*, **14**(8), 1771–1800.

Hinton, G. E., Ghahramani, Z., and Teh, Y. W. (2000). Learning to Parse Images. In S. A. Solla, T. K. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems 12*, pages 463–469. MIT Press, Cambridge, MA.

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

Hyvärinen, A., Hoyer, P. O., and Inki, M. O. (2001). Topographic independent component analysis. *Neural Comput.*, **13**(7), 1527–1558.

Kannan, A., Jojic, N., and Frey, B. J. (2005). Generative model for layers of appearance and deformation. In *AIStats 2005*.

Kannan, A., Winn, J. M., and Rother, C. (2006). Clustering appearance and shape by learning jigsaws. In *NIPS*, pages 657–664.

Karklin, Y. and Lewicki, M. (2009). Emergence of complex cell properties by learning to generalize in natural scenes. *Nature*, **457**, 83–86.

Larochelle, H., Erhan, D., Courville, A., Bergstra, J., and Bengio, Y. (2007). An empirical evaluation of deep architectures on problems with many factors of variation. In

Z. Ghahramani, editor, *Twenty-fourth International Conference on Machine Learning (ICML'2007)*, pages 473–480.

Lee, H., Ekanadham, C., and Ng, A. (2008). Sparse deep belief net model for visual area V2. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*. MIT Press, Cambridge, MA.

Lee, H., Grosse, R., Ranganath, R., and Ng, A. Y. (2009). Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning*, pages 609–616, New York, NY, USA. ACM.

Lewicki, M. and Olshausen, B. (1999). A probabilistic framework for the adaptation and comparison of image codes. *J. Opt. Soc. Am. A*, **16**, 1587–1601.

Olshausen, B. A. and Field, D. J. (1996). Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, **381**(6583), 607–609.

Osindero, S. and Hinton, G. E. (2008). Modeling image patches with a directed hierarchy of Markov random field. In *Neural Information Processing Systems Conference 20*.

Raina, R., Madhavan, A., and Ng, A. Y. (2009). Large-scale deep unsupervised learning using graphics processors. In *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning*, pages 873–880, New York, NY, USA. ACM.

Roth, S. and Black, M. J. (2005). Fields of experts: a framework for learning image priors. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 860–867.

Salakhutdinov, R. (2009). Learning in Markov random fields using tempered transitions. In *Advances in Neural Information Processing Systems 22*, Cambridge, MA. MIT Press.

Salakhutdinov, R. and Murray, I. (2008). On the quantitative analysis of deep belief networks. In *ICML'2008*.

Smolensky, P. (1986). Information processing in dynamical systems: Foundations of harmony theory. In D. E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing*, volume 1, chapter 6, pages 194–281. MIT Press, Cambridge.

Storkey, A. J. and Williams, C. K. I. (2003). Image modeling with position-encoding dynamic trees. *IEEE Trans. Pattern Anal. Mach. Intell.*, **25**(7), 859–871.

Tieleman, T. (2008). Training restricted Boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the International Conference on Machine Learning*, volume 25.

Tu, Z., Chen, X., Yuille, A. L., and Zhu, S.-C. (2005). Image parsing: Unifying segmentation, detection, and recognition. *Int. J. Comput. Vision*, **63**(2), 113–140.

Welling, M., Rosen-Zvi, M., and Hinton, G. E. (2005). Exponential family harmoniums with an application to information retrieval. In *Advances in Neural Information Processing Systems 17*, Cambridge, MA. MIT Press.

Williams, C. and Adams, N. (1999). DTs: Dynamic Trees. In M. Kearns, S. Solla, and D. Cohn, editors, *Advances in Neural Information Processing Systems 11*, pages 634–640. MIT Press.

Williams, C. K. I. and Titsias, M. K. (2004). Greedy learning of multiple objects in images using robust statistics and factorial learning. *Neural Comput.*, **16**(5), 1039–1062.

Winn, J. M. and Jojic, N. (2005). LOCUS: Learning Object Classes with Unsupervised Segmentation. In *ICCV '05: Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, pages 756–763. IEEE Computer Society.

Zhu, S. and Mumford, D. (2006). A stochastic grammar of images. *Found. Trends. Comput. Graph. Vis.*, **2**(4), 259–362.

A Inference and learning in the masked RBM

One of the strengths of RBMs is to have a factorial posterior distribution over the latent variables given the visible ones, making it extremely easy to perform inference. Unfortunately, this is not the case in our model since, even when the mask is known, the latent images are only partially observed, resulting in a non-factorial posterior distribution. Furthermore, the mask is not known for natural images and this needs to be inferred as well. This section explains in detail how to infer all these variables using Gibbs sampling. The mask model we will consider here is the occlusion-based one as the other two can be easily deduced from it. We recall that:

- given a mask $\mathbf{m}$, the log-probability of a joint state $\{\mathbf{v}, \widehat{\mathbf{v}}_{1..K}, \mathbf{h}^{(a)}_{1..K}\}$ is equal to

$$\sum_i \left[\log \delta \left(\widehat{\mathbf{v}}_{m_i,i} = v_i \right) + \sum_j \sum_k f\left(\theta, \widehat{v}_{k,i}, h^{(a)}_{k,j}\right) \right] - \log Z \quad . \tag{14}$$

- the log-probability of a joint state $\{\mathbf{m}, \mathbf{s}_{1..K}, \mathbf{h}^{(s)}_{1..K}, S\}$ is

$$\begin{aligned} \log \quad & P(\mathbf{m}, \mathbf{s}_{1..K}, \mathbf{h}^{(s)}_{1..K}, S) = \log P(S) + \sum_{ijk} \mathbf{s}_{k,i} W_{ij} \mathbf{h}^{(s)}_{k,j} \\ & + \sum_i \left[\log \delta \left(\mathbf{s}_{m_i,i} = 1 \right) + \sum_{k:S(k)<S(m_i)} \log \delta \left(\mathbf{s}_{k,i} = 0 \right) \right] - \log Z \end{aligned} \tag{15}$$

Combining eq. 14 and eq. 15, we get:

$$\begin{aligned} \log \quad & P\left(\mathbf{v}, \widehat{\mathbf{v}}_{1..K}, \mathbf{h}^{(a)}_{1..K}, \mathbf{m}, \mathbf{s}_{1..K}, \mathbf{h}^{(s)}_{1..K}, S\right) = \sum_{ijk} f\left(\theta, \widehat{v}_{k,i}, h^{(a)}_{k,j}\right) + \sum_i \log \delta \left(\widehat{\mathbf{v}}_{m_i,i} = v_i \right) \\ & + \sum_{ijk} \mathbf{s}_{k,i} W^{(s)}_{ij} \mathbf{h}^{(s)}_{k,j} + \sum_i \left[\log \delta \left(\mathbf{s}_{m_i,i} = 1 \right) + \sum_{k:S(k)<S(m_i)} \log \delta \left(\mathbf{s}_{k,i} = 0 \right) \right] \\ & + \log P(S) - \log Z \quad . \end{aligned} \tag{16}$$

A.1 Inference

The joint distribution defined by eq. 16 exhibits several properties:

1. given the latent images $\widehat{\mathbf{v}}_{1..K}$, the distribution over the appearance hidden states $\mathbf{h}^{(a)}_{1..K}$ is factorial

2. given the latent shapes $\mathbf{s}_{1..K}$, the distribution over the shape hidden states $\mathbf{h}^{(s)}_{1..K}$ is factorial

3. given the image patch $\mathbf{v}$, the hidden states $\mathbf{h}^{(a)}_{1..K}$, the hidden states $\mathbf{h}^{(s)}_{1..K}$ and the ordering S, the marginal distribution over the mask $\mathbf{m}$ (when integrating out the latent images $\widehat{\mathbf{v}}_k$ and the latent shapes $\widehat{\mathbf{s}}_{1..K}$) is factorial

4. given the image patch $\mathbf{v}$, the mask $\mathbf{m}$ and the hidden states $\mathbf{h}^{(a)}_k$, the distribution over the latent images $\widehat{\mathbf{v}}_k$ is factorial.

5. given the mask $\mathbf{m}$, the hidden states $\mathbf{h}^{(s)}_k$ and the ordering S, the distribution over the latent shapes $\mathbf{s}_k$ is factorial

Properties 1, 2, 4 and 5 are easily deduced from the form of eq. 16. Let us prove property 3. Given the image patch $\mathbf{v}$, the hidden states $\mathbf{h}^{(a)}_{1..K}$, the hidden states $\mathbf{h}^{(s)}_{1..K}$ and the ordering S, we have

$$\begin{aligned}\log \quad & P\left(\widehat{\mathbf{v}}_{1..K}, \mathbf{m}, \mathbf{s}_{1..K} | \mathbf{v}, \mathbf{h}^{(a)}_{1..K}, \mathbf{h}^{(s)}_{1..K}, S\right) = \sum_{ik} g(\lambda_{ki}\widehat{v}_{k,i}) + \sum_i \log \delta\left(\widehat{\mathbf{v}}_{m_i,i} = v_i\right) \\ & + \quad \sum_{ik} \nu_{ki}\mathbf{s}_{k,i} + \sum_i \left[\log \delta\left(\mathbf{s}_{m_i,i} = 1\right) + \sum_{k/S(k)<S(m_i)} \log \delta\left(\mathbf{s}_{k,i} = 0\right)\right] \\ & - \quad \log Z \qquad .\end{aligned}$$

where $g(\lambda_{ki}\widehat{v}_{k,i}) = \sum_j f\left(\theta, \widehat{v}_{k,i}, h^{(a)}_{k,j}\right)$. From there, we may compute

$$\begin{aligned}\log P\left(\mathbf{m}_i = t | \mathbf{v}, \mathbf{h}^{(a)}_{1..K}, \mathbf{h}^{(s)}_{1..K}, S\right) \quad &= \quad g(\lambda_{ti}v_i) - \log\left(\int_{\widehat{v}_{t,i}} \exp[g(\lambda_{ti}\widehat{v}_{t,i})]\right) \\ &+ \quad \nu_{ti} - \sum_{k/S(k)\leq S(t)} \log\left[1 + \exp(\nu_{ki})\right] - \log Z\end{aligned}$$

which proves the factorial form of the conditional distribution.

This suggests the following Gibbs sampling scheme to infer all the hidden variables given an image $\mathbf{v}$: starting from a random mask $\mathbf{m}$, we iterate over the following steps:

1. given the mask $\mathbf{m}$, we sample the unobserved parts of the latent images $\widehat{\mathbf{v}}_{1..K}$ using block Gibbs sampling (using properties 1 and 4)

2. given the mask $\mathbf{m}$ and the ordering S, we sample the unobserved parts of the latent shapes $\mathbf{s}_{1..K}$ using block Gibbs sampling (using properties 2 and 5)

3. given the latent images $\widehat{\mathbf{v}}_{1..K}$, we sample the appearance hidden units $\mathbf{h}^{(a)}_{1..K}$ (using property 1)

4. given the latent shapes $\mathbf{s}_{1..K}$, we sample the shape hidden units $\mathbf{h}^{(s)}_{1..K}$ (using property 2)

5. given the appearance hidden units $\mathbf{h}^{(a)}_{1..K}$, the shape hidden units $\mathbf{h}^{(s)}_{1..K}$, the image patch $\mathbf{v}$ and the ordering S, we sample a new mask $\mathbf{m}$ (using property 3)

6. given the mask, infer the depth ordering as explained in section B.1

This process is repeated until convergence of the mask. The sampling procedure directly implies that the mask may be different each time. However, in all our experiments, it consistently matched the structure of the shapes in the images.

A.2 Learning

We shall now see how learning of the parameters $W^{(s)}$ and θ can be achieved using the above inference procedure. We need to compute the gradient of the log-probability of an image patch $\mathbf{v}$ with respect to the parameters, that is

$$\frac{\partial \log p(\mathbf{v})}{\partial \theta} = \frac{\partial}{\partial \theta} \log \sum_{\mathbf{m},\widehat{\mathbf{v}}_{1..K},\mathbf{h}^{(a)}_{1..K},\mathbf{s}_{1..K},\mathbf{h}^{(S)}_{1..K},S} P\left(\mathbf{v}, \mathbf{m}, \widehat{\mathbf{v}}_{1..K}, \mathbf{h}^{(a)}_{1..K}, \mathbf{s}_{1..K}, \mathbf{h}^{(S)}_{1..K}, S\right) . \tag{17}$$

Since this can not be computed exactly, we shall use an EM procedure (Dempster *et al.*, 1977). We first derive a variational lower bound of $\log p(\mathbf{v})$:

$$\begin{aligned} \log p(\mathbf{v}) \geq \sum_{\mathbf{m},\widehat{\mathbf{v}}_{1..K},\mathbf{s}_{1..K},S} & Q(\mathbf{m}, \widehat{\mathbf{v}}_{1..K}, \mathbf{s}_{1..K}, S|\mathbf{v}) \\ & \log \sum_{\mathbf{h}^{(a)}_{1..K},\mathbf{h}^{(s)}_{1..K}} P\left(\mathbf{v}, \mathbf{m}, \widehat{\mathbf{v}}_{1..K}, \mathbf{h}^{(a)}_{1..K}\mathbf{s}_{1..K}, \mathbf{h}^{(S)}_{1..K}, S\right) \\ - & H[Q(\mathbf{m}, \widehat{\mathbf{v}}_{1..K}, \mathbf{s}_{1..K}, S|\mathbf{v})] \end{aligned}$$

for any function Q. The bound is tight when $Q(\mathbf{m}, \widehat{\mathbf{v}}_{1..K}, \mathbf{s}_{1..K}, S|\mathbf{v})$ is the true posterior distribution. Since we cannot compute the sum over all masks, all latent images, all latent shapes and all orderings, we will replace it by a sample from the posterior distribution. Therefore, the gradient direction we follow is

$$\Delta\theta \propto \frac{\partial}{\partial \theta} \log \sum_{\mathbf{h}^{(a)}_{1..K},\mathbf{h}^{(s)}_{1..K}} P\left(\mathbf{v}, \widetilde{\mathbf{m}}, \widetilde{\widehat{\mathbf{v}}}_{1..K}, \mathbf{h}^{(a)}_{1..K}, \widetilde{\mathbf{s}}_{1..K}, \mathbf{h}^{(S)}_{1..K}, \widetilde{S}\right) \tag{18}$$

where $\widetilde{\mathbf{m}}$, $\widetilde{\widetilde{\mathbf{v}}}_{1..K}$, $\widetilde{\mathbf{s}}_{1..K}$ and $\widetilde{S}$ are samples from the posterior distribution (obtained using the method described in section A.1). Using more than one sample would reduce noise at the expense of extra computation. In our experiments, we used a single sample and found that learning worked well.

B Depth inference in the occlusion model

B.1 Depth inference for image patches

In order to infer the depth variable S given a mask $\mathbf{m}$ we consider each possible ordering of the K layers explicitly. The mask $\mathbf{m}$ together with a particular occlusion order S defines which shape pixels $s_{k,i}^{(s)}$ are observed and which are unobserved. This is illustrated in Figure 8 in the main text. The likelihood of a particular ordering S is then simply given as the likelihood of all the partially observed shapes $\mathbf{s}_k$ under the shape model:

$$P(S|\mathbf{m}) \propto \prod_{k=1}^{K} \sum_{\{s_{k,i}: i \in U_{S,k}(\mathbf{m})\}} \sum_{\mathbf{h}} p(\mathbf{s}_k, \mathbf{h}). \tag{19}$$

Here, $U_{S,k}(\mathbf{m})$ is the set of all unobserved pixels for shape k given mask $\mathbf{m}$ and ordering S. The set of unobserved pixels $U_{S,k}(\mathbf{m})$ will vary between different orderings S and this is what drives the depth inference.

In practice the sum over unobserved pixels *and* over the latent variables $\mathbf{h}_k^{(s)}$ cannot be computed exactly. We therefore replace the first sum by sampling the unobserved pixels $\{s_{k,i} : i \in U_{S,k}(\mathbf{m})\}$ conditioned on the observed shape pixels for each k and S. Sampling can be done efficiently using several iterations of block Gibbs sampling. This results in "completed" shape images $\hat{s}_k^S$ for which the unnormalized probability under the shape model can be computed efficiently

$$\begin{aligned} p(\hat{\mathbf{s}}_k^S) &= \sum_{\mathbf{h}} p(\hat{\mathbf{s}}_k^S, \mathbf{h}) & (20) \\ &\propto \exp\left(\mathbf{b}^T \hat{\mathbf{s}}_k^S\right) \prod_j \left[1 + \exp\left((\hat{\mathbf{s}}_k^S)^T W_{\cdot j}\right)\right], & (21) \end{aligned}$$

so that we obtain

$$P(S|m, \left\{\hat{\mathbf{s}}_k^S\right\}_{k=1\ldots K}) \propto \prod_{k=1}^{K} p(\hat{\mathbf{s}}_k^S) \tag{22}$$

Note that the completed shape images are different for different S; for plausible orderings the shape model will be able to "fill in" the unobserved pixels to give rise to a shape with a high likelihood which in turn leads to a high probability of the respective ordering. Note further that the shape in the rear-most image is largely determined by preceding layers. In practice we therefore ignore the likelihood of the shape in that layer. Note finally that even though considering each possible ordering S explicitly might seem expensive (the number of possible orderings is factorial in K), for $K \leq 4$ this remains feasible in practice. Given a depth ordering S and the latent states of the K shape RBMs $\{\mathbf{h}_k^{(s)}\}_{k=1\ldots K}$ the conditional probability of the mask is given as

$$P(m_i = t|\{\mathbf{h}_k^{(s)}\}_{k=1\ldots K}, S) = p(s_i = 1|\mathbf{h}_t) \prod_{k \text{ in front of } t} (1 - p(s_i = 1|\mathbf{h}_k)) \quad . \quad (23)$$

This message can be combined with the signal from the appearance models as described in Section A above.

B.2 Depth inference for images

Depth inference at the image level, given a mask image, is performed by determining local depth orderings of overlapping patches. For this purpose each patch is considered in turn and its depth relative to its neighbors is determined, keeping the ordering of its neighbors fixed. For instance, for the experiments with 16×16 pixels patches and $K = 4$ each patch model overlaps partially with eight neighboring patches (so that each pixel is covered by four competing patch models). Thus, for any given patch and a fixed ordering of its eight neighbors, nine different relative depths need to be considered. Each of these relative depths gives rise to a set of unobserved pixels, not only for the patch considered but also for its neighbors. The probability of the different relative depths can be computed in essentially the same way as described in Section B.1 (approximating the sum over unobserved pixels by a sample and then efficiently computing the unnormalized log probability of the completed shape). Note that, for each neighboring patch, the set of unobserved pixels only depends on whether the patch under consideration is in front or behind that neighbor; this considerably reduces the number "shape completions" that need to be considered (two completions per neighboring patch and $N + 1$ for the central patch, where N is the number of neighbors).

In practice, given a mask, we perform one full sweep through the set of patch models, updating the relative depth (and the latent shapes) of each patch with respect to its neighbors once in a random order. Given the resulting depth ordering and the latent states of the shape models the mask can then be updated as in the patch case (cf eq. 23 above).

C Computing the log-probability of image patches under the masked RBM

Due to the number of latent variables involved in the masked RBM, it is impossible to compute the exact log-probability of natural image patches under this model. We may, however, derive a variational lower bound which would allow us to quantify the gains provided by the mask.

C.1 Uniform mask model

We shall begin with the uniform mask model case as this will simplify the equations. From there, it is relatively straightforward to move to the more complex mask models.

$$\begin{aligned}
\log p(\mathbf{v}) &= \log \sum_{\mathbf{m},\widehat{\mathbf{v}}_{1..K},\mathbf{h}^{(a)}_{1..K}} P\left(\mathbf{v},\mathbf{m},\widehat{\mathbf{v}}_{1..K},\mathbf{h}^{(a)}_{1..K}\right) \\
&= \log \sum_{\mathbf{m},\widehat{\mathbf{v}}_{1..K},\mathbf{h}^{(a)}_{1..K}} P\left(\mathbf{v},\mathbf{m},\widehat{\mathbf{v}}_{1..K},\mathbf{h}^{(a)}_{1..K}\right) \frac{Q(\mathbf{m}|\mathbf{v})}{Q(\mathbf{m}|\mathbf{v})} \\
&\geq \sum_{\mathbf{m}} Q(\mathbf{m}|\mathbf{v}) \log P(\mathbf{m}) \sum_{\widehat{\mathbf{v}}_{1..K},\mathbf{h}^{(a)}_{1..K}} P\left(\mathbf{v},\widehat{\mathbf{v}}_{1..K},\mathbf{h}^{(a)}_{1..K}|\mathbf{m}\right) \\
&\quad - \sum_{\mathbf{m}} Q(\mathbf{m}|\mathbf{v}) \log Q(\mathbf{m}|\mathbf{v}) \qquad (24)
\end{aligned}$$

for any function Q, using Jensen's inequality. Let us first rewrite the sum inside the logarithm:

$$P\left(\mathbf{v},\widehat{\mathbf{v}}_{1..K},\mathbf{h}^{(a)}_{1..K}|\mathbf{m}\right) = P\left(\widehat{\mathbf{v}}_{1..K},\mathbf{h}^{(a)}_{1..K}|\mathbf{m}\right) P\left(\mathbf{v}|\widehat{\mathbf{v}}_{1..K},\mathbf{h}^{(a)}_{1..K},\mathbf{m}\right)$$

The second term enforces the constraints described in eq. 14: all configurations which do not match $\widehat{\mathbf{v}}_{m_i,i} = v_i$ for all i have zero probability. Therefore, we only need to com-

pute the sum over the configurations satisfying these constraints. Since these constraints are independent of the $\mathbf{h}_k^{(a)}$, we have

$$P\left(\mathbf{v}|\widehat{\mathbf{v}}_{1..K}, \mathbf{h}_{1..K}^{(a)}, \mathbf{m}\right) = P\left(\mathbf{v}|\widehat{\mathbf{v}}_{1..K}, \mathbf{m}\right)$$

and this distribution is fully concentrated on one point (given the latent images and the mask, there is only one valid image). Furthermore, we have

$$P\left(\widehat{\mathbf{v}}_{1..K}, \mathbf{h}_{1..K}^{(a)}|\mathbf{m}\right) = \prod_{k=1}^{K} P\left(\widehat{\mathbf{v}}_k, \mathbf{h}_k^{(a)}|\mathbf{m}\right) \qquad ,$$

yielding

$$\begin{aligned}
\sum_{\widehat{\mathbf{v}}_{1..K}, \mathbf{h}_{1..K}^{(a)}} P(\mathbf{m}) P\left(\mathbf{v}, \widehat{\mathbf{v}}_{1..K}, \mathbf{h}_{1..K}^{(a)}|\mathbf{m}\right) &= P(\mathbf{m}) \prod_{k=1}^{K} \sum_{\widehat{\mathbf{v}}_k \in C_k, \mathbf{h}_k^{(a)}} P\left(\widehat{\mathbf{v}}_k, \mathbf{h}_k^{(a)}|\mathbf{m}\right) \\
\sum_{\widehat{\mathbf{v}}_{1..K}, \mathbf{h}_{1..K}^{(a)}} P(\mathbf{m}) P\left(\mathbf{v}, \widehat{\mathbf{v}}_{1..K}, \mathbf{h}_{1..K}^{(a)}|\mathbf{m}\right) &= P(\mathbf{m}) \prod_{k=1}^{K} \sum_{\widehat{\mathbf{v}}_k \in C_k} P\left(\widehat{\mathbf{v}}_k|\mathbf{m}\right) \qquad (25)
\end{aligned}$$

where C_k is the set of $\widehat{\mathbf{v}}_k$ matching the constraints imposed by $\mathbf{v}$ and $\mathbf{m}$ (as defined in eq. 14). We recall that the set C_k is the set of all $\widehat{\mathbf{v}}_k$ such that $\widehat{\mathbf{v}}_{k,i} = v_i$ if $m_i = k$. Therefore, we need to sum the probabilities of all visible vectors with a subset of the units being fixed. This can be done using Annealed Importance Sampling (Salakhutdinov and Murray, 2008). Indeed, the conditional distribution over a subset of the visible units given the rest of the other visible units is also an RBM (conditioning on some visible units only modifies the biases of the hidden layer). Given the strong constraint imposed by the observed pixels, the resulting RBM is likely to have a very peaked distribution, making its partition function easy to approximate.

Now that we know how to compute $\sum_{\widehat{\mathbf{v}}_{1..K}, \mathbf{h}_{1..K}^{(a)}} P\left(\mathbf{v}, \mathbf{m}, \widehat{\mathbf{v}}_{1..K}, \mathbf{h}_{1..K}^{(a)}\right)$ for a given $\mathbf{m}$, we need to find the optimal subset of masks to consider (that is, the distribution $Q(\mathbf{m}|\mathbf{v})$).

Let us denote $p_i = P(\mathbf{v}, \mathbf{m}^i)$ for a certain mask configuration $\mathbf{m}^i$ and $q_i = Q(\mathbf{m}^i|\mathbf{v})$. We need to optimize the quantity $D = \sum_i q_i \log p_i - \sum_i q_i \log q_i$ over the q_i's, subject to the constraint $\sum_i q_i = 1$. The optimal solution is given by $q_i = \dfrac{p_i}{\sum_i p_i}$, yielding

$$D = \log \sum_i p_i \qquad (26)$$

We therefore need to find the $\mathbf{m}^i$'s yielding the maximal p_i's. Since $p_i = P(\mathbf{v}, \mathbf{m}^i) = P(\mathbf{v})P(\mathbf{m}^i|\mathbf{v})$, we need to find the modes of the posterior distribution of $\mathbf{m}$ given $\mathbf{v}$. Due to the very constrained nature of the mask, the probability mass is heavily concentrated around a small number of modes, making it possible to achieve a tight bound over the log-probability of an image patch with few masks.

A simpler explanation of this approximation is that we have replaced the quantity $p(\mathbf{v}) = \sum_{\mathbf{m}} p(\mathbf{v}, \mathbf{m})$ by a sum over a subset of the masks. It then becomes clear that this subset needs to include the masks $\mathbf{m}$ for which the quantity $p(\mathbf{v}, \mathbf{m})$ is maximized.

To find the modes of $P(\mathbf{m}|\mathbf{v})$, we shall first do a few iterations (typically twenty) of sampling as described in section A.1, and then replace the third sampling step by a maximization step for a few more iterations (typically ten). We do not perform maximization from the beginning as this often results in finding a poor local optimum.

C.2 Non-uniform mask model

In the case of a non-uniform (occlusion-based) mask model, we have

$$\begin{aligned}
\log p(\mathbf{v}) &= \log \sum_{\mathbf{m}, \widehat{\mathbf{v}}_{1..K}, \mathbf{h}^{(a)}_{1..K}, \mathbf{s}_{1..K}, \mathbf{h}^{(s)}_{1..K}, S} P\left(\mathbf{v}, \mathbf{m}, \widehat{\mathbf{v}}_{1..K}, \mathbf{h}^{(a)}_{1..K}, \mathbf{s}_{1..K}, \mathbf{h}^{(s)}_{1..K}, S\right) \\
&= \log \sum_{\mathbf{m}, \widehat{\mathbf{v}}_{1..K}, \mathbf{h}^{(a)}_{1..K}, \mathbf{s}_{1..K}, \mathbf{h}^{(s)}_{1..K}, S} P\left(\mathbf{v}, \mathbf{m}, \widehat{\mathbf{v}}_{1..K}, \mathbf{h}^{(a)}_{1..K}, \mathbf{s}_{1..K}, \mathbf{h}^{(s)}_{1..K}, S\right) \frac{Q(\mathbf{m}|\mathbf{v})}{Q(\mathbf{m}|\mathbf{v})} \\
&\geq \sum_{\mathbf{m}} Q(\mathbf{m}|\mathbf{v}) \log P(\mathbf{m}) \sum_{\widehat{\mathbf{v}}_{1..K}, \mathbf{h}^{(a)}_{1..K}, \mathbf{s}_{1..K}, \mathbf{h}^{(s)}_{1..K}, S} P\left(\mathbf{v}, \widehat{\mathbf{v}}_{1..K}, \mathbf{h}^{(a)}_{1..K}, \mathbf{s}_{1..K}, \mathbf{h}^{(s)}_{1..K}, S|\mathbf{m}\right) \\
&\quad - \sum_{\mathbf{m}} Q(\mathbf{m}|\mathbf{v}) \log Q(\mathbf{m}|\mathbf{v}) \qquad . \qquad (27)
\end{aligned}$$

Given $\mathbf{m}$, the latent variables may be split in two sets as follows:

$$P\left(\mathbf{v}, \widehat{\mathbf{v}}_{1..K}, \mathbf{h}^{(a)}_{1..K}, \mathbf{s}_{1..K}, \mathbf{h}^{(s)}_{1..K}, S|\mathbf{m}\right) = P(\mathbf{v}, \widehat{\mathbf{v}}_{1..K}, \mathbf{h}^{(a)}_{1..K}|\mathbf{m}) P(\mathbf{s}_{1..K}, \mathbf{h}^{(s)}_{1..K}, S|\mathbf{m}) \qquad (28)$$

and, following the same reasoning as in section C.1, we may compute $P(\mathbf{s}_{1..K}, \mathbf{h}^{(s)}_{1..K}, S|\mathbf{m})$ using AIS.