The Shape Boltzmann Machine: a Strong Model of Object Shape

S. M. Ali Eslami
School of Informatics
University of Edinburgh†
s.m.eslami@sms.ed.ac.uk

Nicolas Heess
Gatsby Unit
University College London†
nheess@gatsby.ucl.ac.uk

John Winn
Microsoft Research
Cambridge
jwinn@microsoft.com

Abstract

A good model of object shape is essential in applications such as segmentation, object detection, inpainting and graphics. For example, when performing segmentation, local constraints on the shape can help where the object boundary is noisy or unclear, and global constraints can resolve ambiguities where background clutter looks similar to part of the object. In general, the stronger the model of shape, the more performance is improved. In this paper, we use a type of Deep Boltzmann Machine [22] that we call a Shape Boltzmann Machine *(ShapeBM) for the task of modeling binary shape images. We show that the ShapeBM characterizes a* strong *model of shape, in that samples from the model look realistic and it can generalize to generate samples that differ from training examples. We find that the ShapeBM learns distributions that are qualitatively and quantitatively better than existing models for this task.*

1. Introduction

Models of the shape of an object play a crucial role in many imaging algorithms, such as those for object detection and segmentation [3, 24, 1, 8], inpainting [6] and graphics [2]. In segmentation, local constraints on the shape, such as smoothness and continuity, can help provide correct segmentations where the object boundary is noisy or lost in shadow. More global constraints, such as ensuring the correct number of parts (legs, wheels, etc.), can resolve ambiguities where background regions look similar to an object part [14]. Shape also plays an important role in generative models of images [21, 8]. In general, the better the model of object shape, the more performance will be improved in these applications.

This paper addresses the question of how to build a *strong* probabilistic model of binary object shapes. We define a strong model as one which meets two requirements:

	Realism		Generalization
	Globally	Locally	
Mean [13]	✓	-	-
Factor Analysis [5]	✓	-	✓
Fragments [3]	-	✓	✓
Grid MRFs/CRFs [20]	-	✓	✓
High-order potentials [17]	limited	✓	✓
Database [12]	✓	✓	-
ShapeBM	✓	✓	✓

Table 1. Comparison of a number of different shape models.

(a) MRF (b) Mean (c) FA (d) ShapeBM

Figure 1. Samples generated by (a) an MRF model of horse shapes, (b) a mean-only model, (c) discrete FA, and (d) the ShapeBM.

1. **Realism** – samples from the model look realistic;
2. **Generalization** – the model can generate samples that differ from training examples.

The first constraint ensures that the model captures shape characteristics at all spatial scales well enough to place probability mass only on images that belong to the 'true' shape distribution. The second constraint ensures that there are no gaps in the learned distribution, i.e. that it also covers novel unseen but valid shapes.

There have been a wide variety of approaches to modeling 2D shape. The most commonly used shape models are grid-structured Markov Random Fields (MRFs) or Conditional Random Fields (CRFs, e.g. [4]). In such models, the pairwise potentials connecting neighboring pixels impose very local constraints like smoothness but are unable to capture more complex properties such as convexity or curvature, nor can they account for longer-range properties. Carefully designed high-order potentials (e.g. [17, 19]) allow particular local or longer-range shape properties to be modeled within an MRF,

†The majority of this work was performed whilst the authors were at Microsoft Research in Cambridge.

but these potentials fall short of capturing *all* such properties so as to make realistic-looking samples.

Other approaches represent shape using a level set or parameterized contour. These have different strengths and weaknesses, but all share the fundamental challenge of imposing sufficient constraints to limit the model to valid shapes while allowing for the right degree of flexibility to capture all possible shapes. For example, a common approach when using a contour (or an image) is to use a mean shape in combination with some principal directions of variation, as captured by a Principal Components Analysis [7] or Factor Analysis [5, 8]. Such models capture the typical global shape of an object and global variations on this shape (such as changes in the aspect ratio of a face). However, they cannot capture multimodal shape distributions, and tend to be poor at learning about local variations which affect only part of the shape (e.g. the angle of a horse's front legs).

Non-parametric approaches employ what is effectively a large database of template shapes [12] or shape fragments [3, 15]. In the former case, because no attempt is made to understand the composition of the shape, it is impossible to generalize to novel shapes not present in the database. In the latter case, the challenge lies in how to compose the shape fragments to form valid shapes. To date, no method has been proposed which can generate a variety of realistic looking whole shapes by composing fragments.

Table 1 and Fig. 1 summarize why these existing approaches do not meet the criteria for a strong shape model.

In this paper, we consider a class of models from the machine learning community, known as Deep Boltzmann Machines (DBMs, [22]). The main contribution of this paper is to show how a strong model of binary shape can be constructed using a form of DBM, which we call the *Shape Boltzmann Machine* (ShapeBM). We demonstrate that a ShapeBM trained on a relatively small dataset is both able to generate realistic samples and to generalize to generate samples that differ from images in the training dataset.

2. Undirected models of shape

In this section we will review several undirected models suitable for modeling binary shape images. We will start with the commonly used grid-structured MRF and describe how it can be modified to form an undirected model known as the Restricted Boltzmann Machine (RBM). We then describe how RBMs can be stacked to form the hierarchical structure of the Deep Boltzmann Machine (DBM).

Grid MRFs: A binary grid-structured MRF defines a distribution over binary images $\mathbf{v}$ whose energy function is:

$$E(\mathbf{v}) = \sum_i f_i(v_i; b_i) + \sum_{(i,j)} f_{ij}(v_i, v_j; w_{ij}), \qquad (1)$$

where i ranges over image pixels, (i, j) ranges over grid edges between pixels i and j and the potentials are parameterized by b_i and w_{ij}. The grid structure of the MRF arises from the pairwise potentials f_{ij} shown in Fig. 2(a). These potentials induce dependencies between neighboring pixels that can favor local shape properties such as connectedness or smoothness. In an attempt to capture more complex or global shape properties, much recent research has focused on constructing higher-order potentials (HOPs), which take the configuration of larger groups of image pixels into account (cf. Sec. 1), but remain computationally tractable. The higher order potentials in [19], for instance, are defined in terms of a set of 'reference patterns' and penalize deviations of groups of pixels from these patterns. Such HOPs can be considered to be introducing an auxiliary hidden variable connected through pairwise potentials to multiple image pixels. The introduction of such hidden variables provides a powerful way to capture and learn complex properties of multiple image pixels. Yet, because the model only contains pairwise potentials, learning and inference remain tractable.

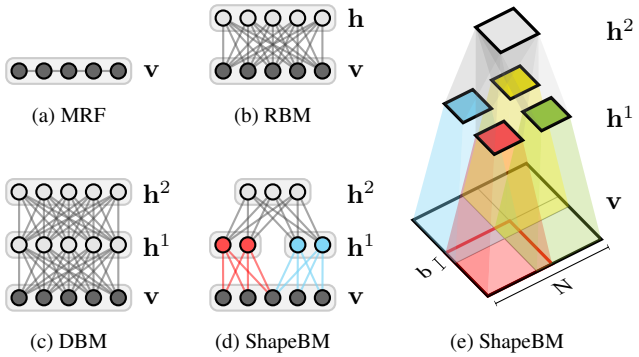

Figure 2. **Undirected models of shape:** (a) 1D slice of a Markov Random Field. (b) Restricted Boltzmann Machine in 1D. (c) Deep Boltzmann Machine in 1D. (d) 1D slice of a Shape Boltzmann Machine. (e) Shape Boltzmann Machine in 2D.

Restricted Boltzmann Machines: A model that makes heavy use of hidden variables is the Restricted Boltzmann Machine (RBM, e.g. [10]). In an RBM, a number of hidden variables $\mathbf{h}$ are used, each of which is connected to all image pixels as shown in Fig. 2(b). However, unlike a grid MRF, there are no direct connections between the image pixels. There are also no direct connections between the hidden variables. Hence, the energy function takes the form:

$$E(\mathbf{v}, \mathbf{h}) = \sum_i b_i v_i + \sum_{i,j} w_{ij} v_i h_j + \sum_j c_j h_j, \qquad (2)$$

where i now ranges over pixels and j ranges over hidden variables. The key points to note are that the potential functions are all simple products and that the only pairwise potentials are those between each visible and each hidden variable. By learning the parameters of the potentials $\{w_{ij}, b_i, c_j\}$, the

model can learn about high-order constraints in the data set.

The distribution over $\mathbf{v}$ is given by marginalizing over the hidden variables: $p(\mathbf{v}) = \sum_{\mathbf{h}} \exp\{-E(\mathbf{v}, \mathbf{h})\}/Z(\Theta)$, where Θ denotes all parameters of the model. This marginalization allows the model to capture high-order dependencies between the visible units. In fact, the hidden units can be summed out analytically [10], giving rise to an alternative formulation of the RBM in terms of high-order potentials that no longer includes latent variables.

Because the RBM has edges only between hidden and visible variables, all hidden units are conditionally independent given the visible units (and vice versa). This property can be exploited to make inference exact and efficient. The conditional probabilities are:

$$p(v_i = 1|\mathbf{h}) = \sigma(\sum_j w_{ij} h_j + b_i), \quad (3)$$

$$p(h_j = 1|\mathbf{v}) = \sigma(\sum_i w_{ij} v_i + c_j), \quad (4)$$

where $\sigma(y) = 1/(1 + \exp(-y))$ is the sigmoid function. This property allows for efficient implementations of block-Gibbs sampling where all $\mathbf{v}$ and all $\mathbf{h}$ are sampled in parallel in an alternating manner, which can be exploited during approximate learning [23].

Deep Boltzmann Machines: RBMs can, in principle, approximate any binary distribution [10], but this can require an exponential number of hidden units and a similarly large amount of training data. The DBM provides a richer model by introducing additional layers of latent variables as shown in Fig. 2(c). The additional layers capture high-order dependencies between the hidden variables of previous layers and so can learn about complex structure in the data using relatively few hidden units. The energy of a DBM with two layers of latent variables is given by:

$$E(\mathbf{v}, \mathbf{h}^1, \mathbf{h}^2) = \sum_i b_i v_i + \sum_{i,j} w^1_{ij} v_i h^1_j + \sum_j c^1_j h^1_j + \sum_{j,k} w^2_{jk} h^1_j h^2_k + \sum_k c^2_k h^2_k. \quad (5)$$

Although exact inference is no longer possible in this model, the conditional distributions $p(\mathbf{v}|\mathbf{h}^1)$, $p(\mathbf{h}^1|\mathbf{v}, \mathbf{h}^2)$, and $p(\mathbf{h}^2|\mathbf{h}^1)$ remain independent due to the layering (taking forms analogous to Eqs. 3, 4). This allows for computationally efficient inference, either by layerwise block-Gibbs sampling from the posterior $p(\mathbf{h}^1, \mathbf{h}^2|\mathbf{v})$ (Fig. 3), or by using a mean field procedure with a fully factorized approximate posterior as described in [22]. The layering further admits a layer-wise pre-training procedure that makes it less likely that learning will get stuck in local optima. Hence the DBM is both a rich model of binary images and a tractable one.

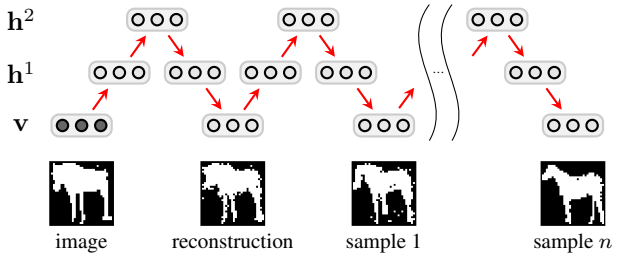

Figure 3. **DBM MCMC.** Block-Gibbs MCMC sampling scheme, in which $\mathbf{v}$, $\mathbf{h}^1$ and $\mathbf{h}^2$ variables are sampled in turn. Note that each sample of $\mathbf{h}^1$ is obtained conditioned on the current state of $\mathbf{v}$ *and* $\mathbf{h}^2$. For sufficiently large values of n, sample n will be uncorrelated with the original image.

3. The Shape Boltzmann Machine

RBMs and DBMs are powerful generative models, but also have many parameters. Since they are typically trained on large amounts of unlabeled data (thousands or tens of thousands of examples), this is usually less of a problem than in supervised settings. Segmented images, however, are expensive to obtain and datasets are typically small (hundreds of examples). In order to learn a model that accurately captures the properties of binary shapes, but also generalizes even when trained on small datasets, we use a form of DBM but additionally impose carefully chosen connectivity and capacity constraints (in a similar vein to [16, 18]).

The ShapeBM used below has two layers of latent variables: $\mathbf{h}^1$ and $\mathbf{h}^2$. The visible units $\mathbf{v}$ are the pixels of a binary image of size $N \times N$. In the first layer we enforce local receptive fields by connecting each hidden unit in $\mathbf{h}^1$ only to a subset of the visible units, corresponding to one of four square patches, as shown in Fig. 2(d,e). Each patch overlaps its neighbor by b pixels and so has a side length of $N/2 + b/2$. We furthermore share weights between the four sets of hidden units and patches. These modifications reduce the number of first layer parameters by a factor of about 16 which reduces the amount of data needed for training by a similar factor. At the same time these modifications take into account two important properties of shapes: first, the restricted receptive field size reflects the fact that the strongest dependencies between pixels are typically local, while distant parts of an object often vary more independently (the small overlap allows boundary continuity to be learned primarily at the lowest layer); second, weight sharing takes account of the fact that many generic properties of shapes (e.g. smoothness) are independent of the image position.

For the second layer we choose full connectivity between $\mathbf{h}^1$ and $\mathbf{h}^2$, but restrict the relative capacity of $\mathbf{h}^2$: we use 4×500 hidden units for $\mathbf{h}^1$ vs. 50 or 100 for $\mathbf{h}^2$ in our single class experiments. While the first layer is primarily concerned with generic, local properties, the role of the

second layer is to impose global constraints, e.g. with respect to the class of an object shape or its overall posture. The second layer mediates dependencies between pixels that are far apart (not in the same local receptive field), but these dependencies will be weaker than between nearby pixels that share first-level hidden units. Limiting the capacity of the second-layer encourages this separation of concerns and helps to prevent the model from overfitting to small training sets. Note that this is in contrast to [22] which use a top-most layer that is at least as large as all of the preceding layers.

Learning: Learning of the model involves maximizing $\log p(\mathbf{v};\Theta)$ of the observed data $\mathbf{v}$ with respect to its parameters $\Theta = \{\mathbf{b}, W^1, W^2, \mathbf{c}^1, \mathbf{c}^2\}$ (cf. Eq. 5). This is difficult for three reasons: (1) the intractability of the normalization constant Z (which depends on the parameters); (2) the presence of latent variables; and (3) the tendency of learning to get stuck in poor local optima. The procedure proposed in [22] minimizes these difficulties and we follow it closely.

Learning proceeds in two phases. In the *pre-training* phase we greedily train the model bottom up, one layer at a time. The purpose of this phase is to find good initial values for all parameters of the model. We begin by training an RBM on the observed data using stochastic maximum likelihood learning (SML, also referred to as 'persistent CD', [23, 22]). The number of hidden units of this RBM is the same as the size of $\mathbf{h}^1$ in the full ShapeBM model and it obeys the same connectivity constraints as the ShapeBM's first layer. Once this RBM is trained, we infer the conditional mean of the hidden units using Eq. 4 for each training image. The resulting vectors then serve as the training data for a second RBM with the same number of hidden units as $\mathbf{h}^2$, which is again trained using SML.

We use the parameters of these two RBMs to initialize the parameters of the full ShapeBM model as described in [22]. In the second phase we perform approximate stochastic gradient ascent in the likelihood of the full model to fine-tune the parameters in an expectation-maximization-like scheme. This involves the same sample-based approximation to the gradient of the normalization constant used for learning the RBMs [23, 22], as well as a mean-field approximation to the posterior $p(\mathbf{h}^1, \mathbf{h}^2|\mathbf{v})$ of training images. This joint training is essential to separate out learning of local and global shape properties into the two hidden layers.

4. Experiments

We performed both qualitative and quantitative experiments to assess whether the ShapeBM can act as a strong model of object shape.

4.1. Weizmann horses

The first dataset we investigated was the Weizmann horse dataset [3] which contains 327 images, all of horses facing to the left, but in a variety of poses. The binary images are cropped and normalized to 32×32 pixels (see Fig. 4(a)). This dataset is challenging, because in addition to their overall pose variation, the positions of the horses' heads, tails and legs change considerably from image to image. Compared to the amount of variability seen in the data, the number of training images is relatively small.

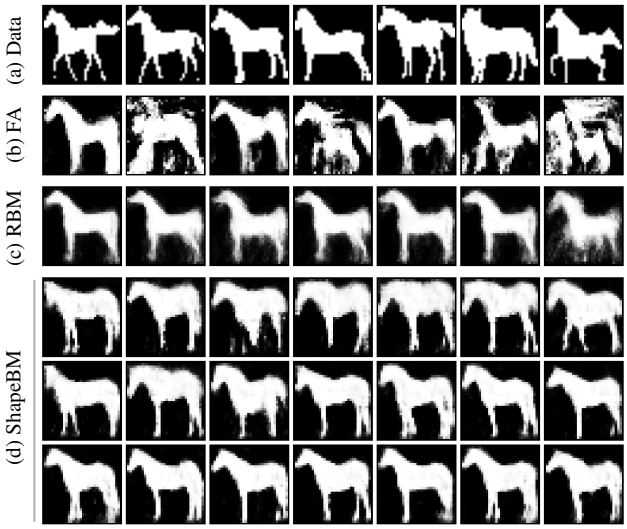

Figure 4. **Sampled shapes.** (a) A selection of images from the Weizmann horse dataset. (b) A collection of samples from a discrete Factor Analysis model. The Gaussianity assumption forces the model to allocate probability mass to unlikely horse shapes. (c) Samples from an RBM. (d) Samples from a ShapeBM. The model generates samples of varying pose, with the correct numbers of legs and details are preserved (samples are arranged l-r, u-d in decreasing order of generalization).

We trained a ShapeBM with overlap $b = 4$, and 2000 and 100 units for $\mathbf{h}^1$ and $\mathbf{h}^2$ respectively. The first layer was pre-trained for 3000 epochs (iterations) and the second layer for 1000 epochs. After pre-training, joint training was performed for 1000 epochs. Our MATLAB implementation completed training in around 4 hours, running on a dual-core, 3GHz PC with 4GB of memory.

For comparison, we trained a Factor Analysis (FA) model with 10 latent dimensions, and an RBM with 500 hidden units on the same data[1]. The FA model was modified to work on discrete binary images, similarly to the Clipped Factor Analysis model described in [5].

Realism: To assess the Realism requirement, we sampled a set of shapes[2] from each model, as shown in Fig. 4. FA

[1] We obtained the best results with these settings of the parameters.

[2] In the sampling figures, we display the (grayscale) *conditional probability* of each pixel given a particular hidden configuration. Binary samples can be generated per-pixel from a Bernoulli distribution where the gray level specifies the distribution mean.

effectively defines a Gaussian distribution over the image pixels and is thus inherently unimodal. In order to account for the diversity of shapes in the training data it is therefore forced to allocate probability mass to images that do not correspond to realistic horse shapes, as shown in Fig. 4(b).

By contrast, the RBM can, in principle, account for multi-modal data and could thus assign probability mass more selectively. However, as the samples of Fig. 4(c) indicate, the model has also failed to learn a good model of the variability of horse shapes – the samples are mostly of the same pose, and details of the shape are lost when the pose changes. These problems are symptomatic of training with insufficient data. Increasing the number of hidden units in the hope of learning more local filters did not solve the problem, confirming that the lack of data is the issue[3].

We now consider the samples from the ShapeBM in Fig. 4(d). These were generated using the scheme outlined in Fig. 3. First, we note that the model generates natural shapes from a variety of poses. Second, we observe that details such as legs are preserved and remain sharply defined in the samples. Third, we note that the horses have the correct number of legs. Finally, we note that the patch overlap ensures seamless connections between the four quadrants of the image. Indeed, horse samples generated by the model look sufficiently realistic that we consider the model to have fulfilled the Realism requirement.

Generalization: We next investigated to what extent the ShapeBM meets the Generalization requirement, to ensure that the model has not simply memorized the training data. In Fig. 5 we show the difference between the sampled shapes from Fig. 4(d) and their closest images in the training set. We use the Hamming distance between training images and a thresholded version of the conditional probability (> 0.3), as the similarity measure[4]. Red indicates pixels that are in the sample but not in the closest training image, and yellow indicates pixels in the training image but not in the sample. Fig. 5 shows that the model generalizes to realistic horse shapes that it has not encountered in the training set.

Another way to diagnose the generalization behavior of the models is to inspect their first layer weight matrices. Each column in W corresponds to a 'filter' that is associated with the activation of one of the hidden units. As shown in Fig. 6, the filters for the FA and RBM have only global structure. This means they are unable to combine local filters to generate novel horse shapes. In contrast, because spatial locality and parameter-sharing are built into the ShapeBM, it learns general-purpose filters that allow it to generalize factorially from the training examples.

[3] An RBM with similar connectivity constraints as the first layer of the ShapeBM has fewer parameters than a fully connected RBM and thus suffers less from overfitting, but without the second layer it fails to account for global constraints on the shape.

[4] This measure was found to retrieve the visually most similar images.

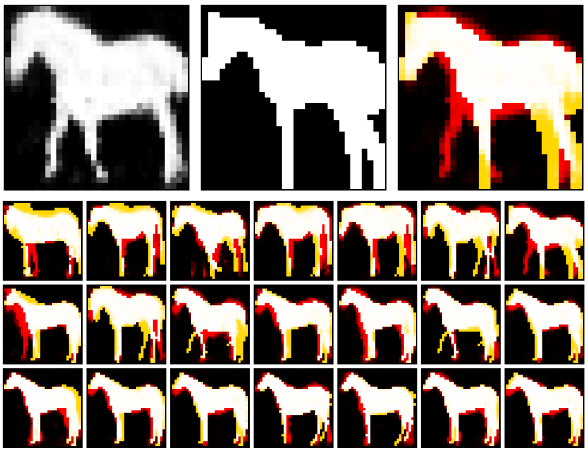

Figure 5. **Generalization.** *Top:* A sample from the ShapeBM, the closest image in the training dataset to the generated sample, and the difference between the two images. Red pixels have been generated by the sample but are absent in the training image; yellow pixels are present in the training image but absent in the sample. The model has generalized to an unseen, but realistic horse shape. *Bottom:* Generalizations made in each of the samples in Fig. 4(d).

Learned invariance: To tease out the kinds of information captured at the different levels of the model, in Fig. 7 we plot two sets of samples from an alternative sampling scheme. In these chains we only iterate between sampling $\mathbf{v}$ and $\mathbf{h}^1$, and we keep the $\mathbf{h}^2$ variables fixed to values sampled using one of the training images. We observe that in doing so we fix the horse's pose, but since $\mathbf{h}^1$ changes from sample to sample the position of its legs and other small details vary. This suggests that the highest layer in the model predominantly captures global information and has learned to be *invariant* to small-scale changes in shape. This automatic, implicit, separation of large-scale and small-scale statistics is fundamental to the operation of the model.

Shape completion: We further assessed both the realism and generalization capabilities of the ShapeBM by using it to perform shape completion, where the goal is to generate

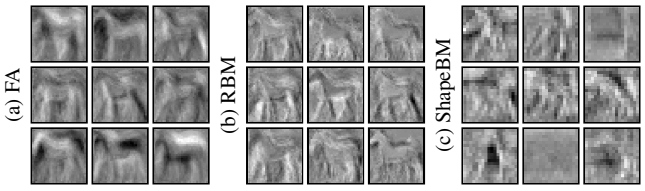

Figure 6. **First layer example weights.** (a) Weights learned by the FA model capture only global modes of variability. (b) Weights learned by the RBM also fail to capture local modes of variation. (c) General, more local filters learned by a ShapeBM (18×18).

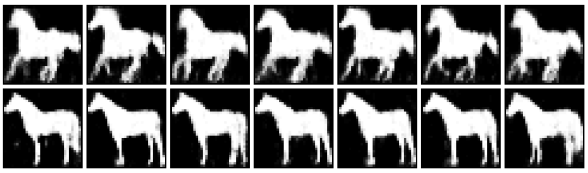

Figure 7. **Learned invariance.** Sampling chains are run for two fixed, but different, configurations of $\mathbf{h}^2$. The horse's pose remains fixed, but configurations of legs, and neck and back positions vary. This suggests that the highest layer in the model predominantly captures high-level pose information.

likely configurations of pixels for a missing region of the shape, given the rest of the shape. To perform completion, we sample as before, but every time $\mathbf{v}$ is sampled we 'clamp' the observed pixels of the image to their given values. Since the model specifies a *distribution* over the missing region, multiple such samples capture the variability of possible solutions that exist for any given completion task. In Fig. 8 we show how the samples become more constrained as the missing region shrinks. Fig. 9 shows completions of rectangular regions of images that the model had not seen during training. Despite the large sizes of the missing portions, and the horses' varying poses, completions look realistic.

The ShapeBM's ability to do shape completion suggests applications in a computer graphics setting. Sampled completions can be constrained in real-time by simply clamping certain pixels of the image. In Fig. 10(a) we show snapshots of a graphical user interface in which the user modifies a horse silhouette with a digital brush. The model's ability to generalize enables it to generate samples that satisfy the user's constraints. The model's accurate knowledge about horse shapes ensures that the samples remain realistic. As shown, a database-driven approach can fail to find shapes that match the constraints. The same sampling technique can also be used to generate complete horse silhouettes in different poses given simple stick figures provided by the

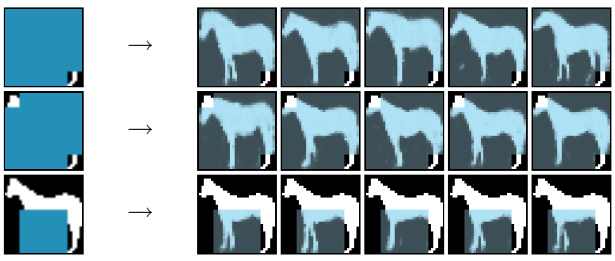

Figure 8. **Shape completion variability.** Blue in the first column indicates the missing regions. The samples highlight the *variability* in possible completions captured by the model. As the missing region shrinks, the samples become more constrained.

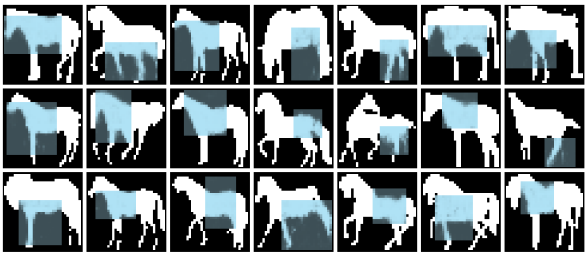

Figure 9. **Image completion.** The ShapeBM completes rectangular imputations of random size on images not seen during training.

	Mean	RBM	FA	ShapeBM
Score	-50.72	-47.00	-40.82	$\mathbf{-28.85}$

Table 2. **Imputation scores.**

user (see Fig. 10(b)). This GUI and a video showing its use may be downloaded from http://bit.ly/ShapeBM.

Quantitative comparison: In order to compare the performance of the models quantitatively, we introduce what we will refer to as an 'imputation score' for the shape completion task as a measure of the strength of a model. We collect 25 additional horse silhouettes from the web, and divide each into 9 segments. In each imputation test, we remove one of the segments and estimate the conditional probability of that segment under the model, given the remaining 8 segments. The log probabilities are then averaged across the different segments and images to give the score. Log-scores, for a particular image and segment, are approximated via sampling. We draw configurations of the latent variables from the posterior given the observed part of the image and then evaluate the conditional probability of the true configuration of the unobserved segment given the latent variables, i.e. $p(\mathbf{v}_u|\mathbf{v}_o) \approx \sum_s p(\mathbf{v}_u|\mathbf{h}^s)/S$, where u and o indicate the set of unobserved/observed pixels, and $\mathbf{h}^s$ are samples from $p(\mathbf{h}|\mathbf{v}_o)$ obtained via MCMC. A high score in this test indicates both the realism of samples and the generalization capability of a model, since models that do not allocate probability mass on good shapes (from the 'true' generating distribution of horses) and models that waste probability mass on bad shapes are both penalized. As the results show in Table 2, the ShapeBM significantly outperforms the other models at this task.

4.2. Caltech-101 motorbikes

In the previous section we demonstrated that the ShapeBM model can learn a strong model for horses. To verify that the model works not just for horses, and to test on higher-resolution images, we trained a model on 798 motorbike silhouettes from the Caltech-101 dataset [9]. Here, the binary images are cropped and normalized to 64×64

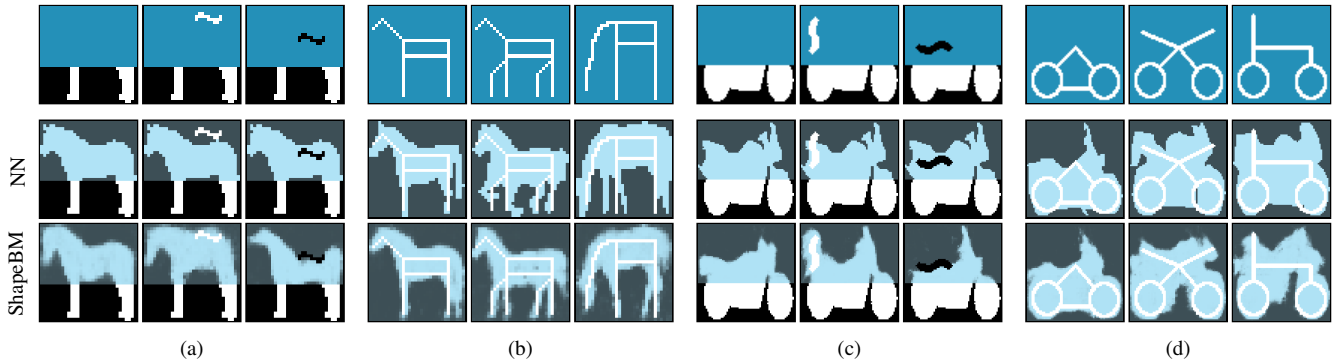

Figure 10. **Constrained shape completion.** Missing regions (blue pixels, top row) are completed using the ShapeBM and by finding the closest match (NN) to the prescribed pixels in the training data. (a) The horse's back is pulled up by the ShapeBM using an appropriate 'on' brush. Notice how the stomach moves up and the head angle changes to maintain a valid shape. The horse's back is then pushed down with an 'off' brush. (b) Given only minimal user input, the model completes the images to generate realistic horse shapes. (c,d) Motorbikes (at 64×64). In most cases, the nearest neighbor method fails to find a suitable training image to satisfy the constraints.

pixels (see Fig. 11(a)). We trained a ShapeBM with overlap $b = 4$, and 1200 and 50 units for $\mathbf{h}^1$ and $\mathbf{h}^2$ respectively, using the same schedule as before. Fig. 11 displays samples from the model, as well as shape completions on unseen data. The model generalizes from the training data-points in non-trivial ways whilst maintaining validity of the overall object shape. Image completions look natural and connect smoothly at the boundaries. We demonstrate constrained image completion in Fig. 10(c,d) and note that the model generates conformant shapes that align neatly with the fixed portion of the image. The database-driven approach can fail to find suitable images in the training dataset.

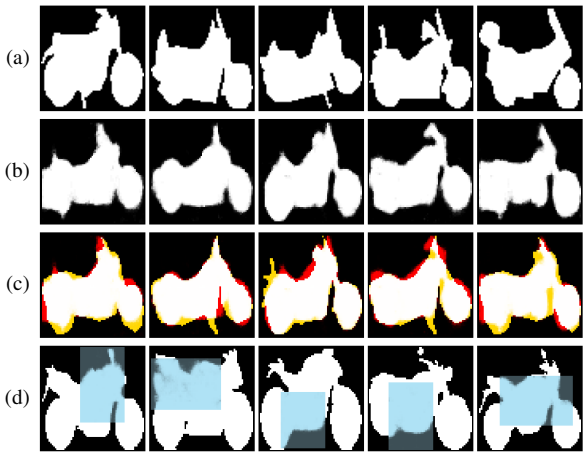

Figure 11. **Results on Caltech-101 motorbikes.** (a) A selection of images from the dataset. (b) A chain of samples generated by the ShapeBM (at 64×64). (c) The model generalizes from training examples in non-trivial ways, whilst maintaining overall motorbike look-and-feel. (d) Completion of images not seen during training.

4.3. Multiple object categories

Class-specific shape models are appropriate if the class is known, but for segmentation/detection applications the object class may not be known. To this end, we investigated if the ShapeBM can be used to learn shapes of multiple object classes simultaneously. For this purpose, we trained a ShapeBM on a combination of the Weizmann data and 3 other animal categories from Caltech-101 [9]. In addition to 327 horse images, the dataset contains images of 68 dragonflies, 78 llamas and 59 rhinos (for a total of 531 images). The images are cropped and normalized to 32×32 pixels. A ShapeBM with $b = 4$, and 2000 and 400 units for $\mathbf{h}^1$ and $\mathbf{h}^2$ was jointly trained *without* information about image class.

In our experiments we have found that the ShapeBM still learns a strong model, as demonstrated by Fig. 12. It would be informative to know if ShapeBM's unsupervised learning procedure has led it to discover the underlying grouping of the shapes into categories. In order to do this, we compute average inter- and intra-class distances of all training instances, both in data-space ($\mathbf{v}$) and in latent-space ($\mathbf{h}^2$). In Fig. 13(a) we plot the ratio of these distances for the four classes. These results suggest that the ShapeBM latent representation groups the shapes from each category much more closely than they are in pixel-space.

We also tested how well the model discovered object categories by using it to classify in a setting with very few labeled examples. We trained a generalized linear model (GLM) using the `glmnet` algorithm [11] on between $T = 1 \ldots 20$ randomly selected images of each category and tested on $59 - T$ images per category, averaging over 100 runs. We find that despite its smaller size, given only a few training examples, the latent $\mathbf{h}^2$ is most discriminative (see Fig. 13(b)). After just one labeled example per category, classification accuracy is 56.0% using $\mathbf{h}^2$ vs. just 36.8% using $\mathbf{v}$.

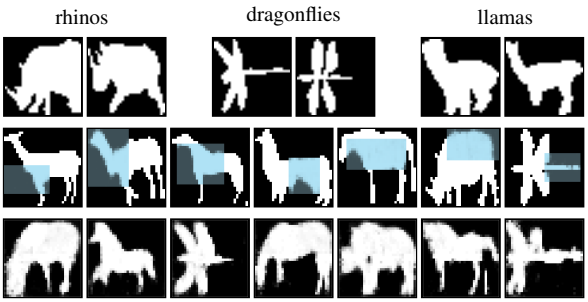

Figure 12. **Multiple object categories.** *Top:* A selection of images from the augmented dataset. *Middle:* The model simultaneously identifies the object class and fills in the missing image region. *Bottom:* Samples from a single tempered chain.

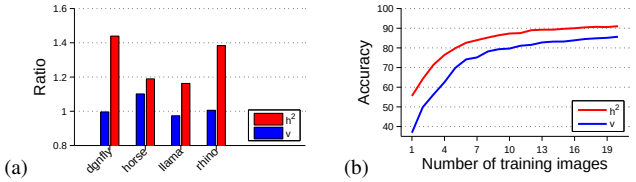

Figure 13. (a) Inter- and intra-class distance ratios (values > 1 indicate that inter-class distances are larger). (b) GLM classification.

5. Conclusions and future work

In this paper we have shown how the ShapeBM can learn high quality probability distributions over object shapes in terms of both realism of samples from the distribution and generalization to new examples of the same shape class.

In future, we plan to investigate how the model scales to higher-resolution images and how it can be extended to reason about multiple occluding objects (in a similar vein to [21]). We believe that such a model would be suited for use as a prior for tasks like the PASCAL segmentation challenge.

Acknowledgements: AE has received funding from the Carnegie Trust, the SORSAS scheme, and the IST Programme of the European Community under the PASCAL2 Network of Excellence (IST-2007-216886). NH has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 231495 and from the Gatsby Charitable foundation.

References

[1] B. Alexe, T. Deselaers, and V. Ferrari. ClassCut for unsupervised segmentation. In *ECCV*, pages 380–393, 2010. 1

[2] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis. SCAPE: shape completion and animation of people. *SIGGRAPH*, 24(3):408–416, 2005. 1

[3] E. Borenstein, E. Sharon, and S. Ullman. Combining Top-Down and Bottom-Up Segmentation. In *CVPR Workshop on Perceptual Organization in Computer Vision*, 2004. 1, 2, 4

[4] Y. Boykov and M.-P. Jolly. Interactive Graph Cuts for Optimal Boundary & Region Segmentation of Objects in N-D images. In *ICCV*, pages 105–112, 2001. 1

[5] T. Cemgil, W. Zajdel, and B. Krose. A Hybrid Graphical Model for Robust Feature Extraction from Video. In *CVPR*, pages 1158–1165, 2005. 1, 2, 4

[6] T. F. Chan and J. Shen. Nontexture Inpainting by Curvature-Driven Diffusions. *JVCI*, 12(4):436–449, 2001. 1

[7] T. Cootes, C. Taylor, D. H. Cooper, and J. Graham. Active shape models—their training and application. *Computer Vision and Image Understanding*, 61:38–59, 1995. 2

[8] S. M. A. Eslami and C. K. I. Williams. Factored Shapes and Appearances for Parts-based Object Understanding. In *BMVC*, pages 18.1–18.12, 2011. 1, 2

[9] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples. In *CVPR Workshop on Generative-Model Based Vision*, 2004. 6, 7

[10] Y. Freund and D. Haussler. Unsupervised learning of distributions on binary vectors using two layer networks. Technical Report UCSC-CRL-94-25, UCSC, 1994. 2, 3

[11] J. Friedman, T. Hastie, and R. Tibshirani. Regularization Paths for Generalized Linear Models via Coordinate Descent. *JSS*, 33(1):1–22, 2010. 7

[12] D. Gavrila. An Exemplar-Based Approach to Hierarchical Shape Matching. *PAMI*, 29:1408–1421, 2007. 1, 2

[13] N. Jojic and Y. Caspi. Capturing Image Structure with Probabilistic Index Maps. In *CVPR*, pages 212–219, 2004. 1

[14] N. Jojic, A. Perina, M. Cristani, V. Murino, and B. Frey. Stel component analysis: Modeling spatial correlations in image class structure. In *CVPR*, pages 2044–2051, 2009. 1

[15] P. Kumar, P. Torr, and A. Zisserman. OBJ CUT. In *CVPR*, pages 18–25, 2005. 2

[16] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng. Convolutional DBNs for Scalable Unsupervised Learning of Hierarchical Representations. In *ICML*, pages 609–616, 2009. 3

[17] S. Nowozin and C. Lampert. Global connectivity potentials for random field models. In *CVPR*, pages 818–825, 2009. 1

[18] M. Ranzato, V. Mnih, and G. E. Hinton. How to Generate Realistic Images Using Gated MRFs. In *NIPS*, 2010. 3

[19] C. Rother, P. Kohli, W. Feng, and J. Jia. Minimizing sparse higher order energy functions of discrete variables. In *CVPR*, pages 1382–1389, 2009. 1, 2

[20] C. Rother, V. Kolmogorov, and A. Blake. "GrabCut": interactive foreground extraction using iterated graph cuts. *SIGGRAPH*, 23:309–314, 2004. 1

[21] N. L. Roux, N. Heess, J. Shotton, and J. Winn. Learning a Generative Model of Images by Factoring Appearance and Shape. *Neural Computation*, 23(3):593–650, 2011. 1, 8

[22] R. Salakhutdinov and G. Hinton. Deep Boltzmann Machines. In *AISTATS*, volume 5, pages 448–455, 2009. 1, 2, 3, 4

[23] T. Tieleman. Training restricted Boltzmann machines using approximations to the likelihood gradient. In *ICML*, pages 1064–1071, 2008. 3, 4

[24] J. Winn and N. Jojic. LOCUS: Learning object classes with unsupervised segmentation. In *ICCV*, 2005. 1