A Hand Shape Recognizer from Simple Sketches

Xiaolong Zhu, Ruoxin Sang, Xuhui Jia, Kwan-Yee K. Wong
Department of Computer Science
The University of Hong Kong
Pokfulam Road, Hong Kong
{xlzhu,rxsang,xhjia,kykwong}@cs.hku.hk

***Abstract*—Hand shape recognition is one of the most important techniques used in human-computer interaction. However, it often takes developers great efforts to customize their hand shape recognizers. In this paper, we present a novel method that enables a hand shape recognizer to be built automatically from simple sketches, such as a "stick-figure" of a hand shape. We introduce the Hand Boltzmann Machine (HBM), a generative model built upon unsupervised learning, to represent the hand shape space of a binary image, and formulate the user provided sketches as an initial guidance for sampling to generate realistic hand shape samples. Such samples are then used to train a hand shape recognizer. We evaluate our method and compare it with other state-of-the-art models in three aspects, namely i) its capability of handling different sketch input, ii) its classification accuracy, and iii) its ability to handle occlusions. Experimental results demonstrate the great potential of our method in real world applications.**

***Index Terms*—hand shape, sketch, generative model**

I. Introduction

Hand shape recognition is an essential technique in natural user interface, particularly in sign language recognition, virtual reality and computer entertainment [1]. In recent years, the great progress in depth sensors (*e.g.*, Kinect [2] and Intel Gesture Camera [3]) has made their prices more affordable, which in turn increases their utilities in hand shape recognition in daily life. By using a depth camera, it is no longer difficult to segment and track a hand from an image sequence. A binary hand shape segmentation can often be extracted from a depth image as a hand shape representation [4], [5], which serves as an input for hand shape recognition. However, it is a well-known fact that the hand shape of a single gesture may vary among different people due to different applications or cultures. A developer of a hand shape recognizer therefore needs to collect a huge amount of samples from many subjects in order to train a robust recognizer for his application, and he may need to repeat this process if the accuracy of the recognizer is not satisfactory or if new hand shapes are needed. This often takes the developers a lot of time to refine their recognizers iteratively. In addition, many developers nowadays come from the design community, and they do not necessarily have the background in computer vision and pattern recognition. Hence it may be a great barrier for them to manage the training data and train a customized hand shape recognizer by themselves.

978-1-4799-0883-7/13/$31.00 ©2013 IEEE

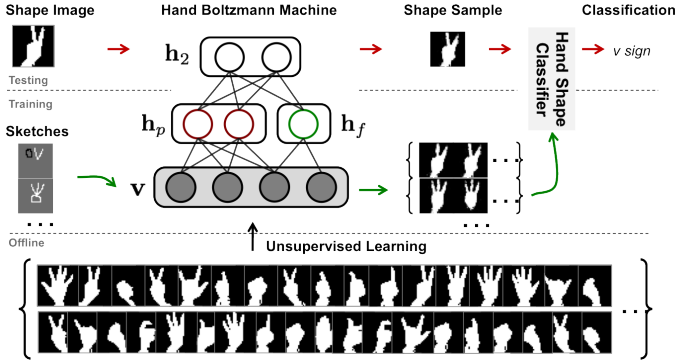

Fig. 1. Illustration of our method. The hand shape model is firstly learned from the dataset, and the images are sampled from this model to train a hand shape classifier. During testing, a shape image is first preprocessed through HBM and then used as the input for the shape classifier.

From a probabilistic perspective, acquiring samples for different hand shapes can be viewed as drawing samples from the conditional probability over different hand shape configurations. This suggests that a sampling method can be used to collect hand shape samples for training if the distribution of the hand shape image is known. In other words, if we can build up a probabilistic hand shape model, it is not necessary to collect a large amount of samples for training. Since the probabilistic hand shape model models the joint probability of pixels in a hand shape image, a complete hand shape image can be sampled even from partially observed data [6]. This suggests that intuitive drawings like simple sketches of a hand shape can be used to generate similar hand shape images.

In this paper, we address the following problem: *is it possible for developers to use simple sketches of hand shapes instead of a large collection of training data to develop their customized hand shape recognizers?* To solve this problem, we propose a mid-level representation to model the hand shape images in a probabilistic fashion, which allows generation of new data that are different from the training samples (see Fig. 1). Hence it becomes possible for developers to customize their hand shape recognizers by simply sketching the hand shapes of their interests.

Our method is inspired by the idea of unsupervised learning for image completion. Successful applications can be found in face completion [6], shape completion [7], *etc*. On the other hand, sketch-based systems show great potential for describing

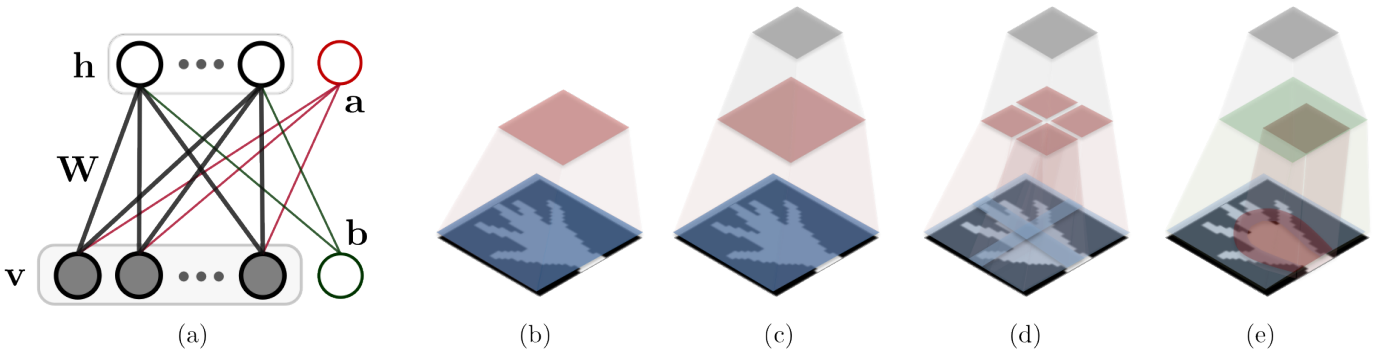

Fig. 2. (a) The structure of RBM. (b)-(e) The illustration of different undirected models for a hand shape image. From left to right, they are RBM, DBM, ShapeBM and our proposed HBM. Each node in the visual layer corresponds to a pixel in the image.

and retrieving images from a large database [8]. We combine these ideas to enable users to generate hand shape images by sketching the hand shape in their minds. Our method consists of two components: firstly, a probabilistic hand shape model, namely Hand Boltzmann Machine (HBM), is introduced to represent a binary hand shape image, and secondly, the sketch by a user is modeled as constraints in generating samples for training a hand shape recognizer. Given an input hand shape image, it is first preprocessed by the model and the output will serve as the input for hand shape recognition. By formulating the problem in this way, we make the following contributions:

- We introduce the Hand Boltzmann Machine, which is tailored for the structure of a hand shape image, to generate realistic sample images.
- It is flexible and intuitive for developers to develop their hand shape recognizers using simple sketches instead of collecting tens of thousands of hand shape images in different hand shape configurations;
- Our method can deal with occlusions and remove noisy pixels by preprocessing the partially observed hand shape images using the learned model.
- Customization of a hand shape recognizer is separated from data acquisition, so it is much easier for developers to collect (generate) hand shapes for training. This has great potential for industrial use: data providers may focus on making good models while developers can customize their classifiers in less time;

II. Literature Review

There have been plenty of methods modeling 2D shape images. Shape templates [9] and shape fragments [10], [11] are two common non-parametric methods for a large database of image shapes. These methods cannot generate novel and realistic shapes as there is no explicit formulation to validate the composition of different shapes or shape fragments. On the other hand, probabilistic models such as MRFs and CRFs [12] model a shape image as grid-structured potentials of connected pixels. However, they cannot capture higher level properties, such as curvature and convexity, of the shapes although local smoothness is explicitly constrained.

Recently, deep architectures have shown great potentials in capturing complex global shape properties. By introducing the layer-based structure, such information can be represented in the nodes of the higher layer. In the followings, we first review several representatives of these models (See Fig. 2).

- **Restricted Boltzmann Machine**. It is a two-layered undirected model with a visual layer $\mathbf{v}$ representing binary image pixels and a hidden layer $\mathbf{h}$ representing the latent correlation among these visual nodes. The energy function is defined as

$$E(\mathbf{v},\mathbf{h}) = -\mathbf{a}^\top\mathbf{v} - \mathbf{b}^\top\mathbf{h} - \mathbf{v}^\top\mathbf{W}\mathbf{h}, \tag{1}$$

  with the parameters $\{\mathbf{a},\mathbf{b},\mathbf{W}\}$ as illustrated in Fig. 2(a). The distribution over $\mathbf{v}$ is obtained by summing up all over the hidden variables, *i.e.*, $P(\mathbf{v}) = \frac{1}{Z}\sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}$, where Z is a normalization term.
- **Deep Boltzmann Machine**. In order to capture high level complex correlation between hidden variables, a deeper architecture is often pursued. Such a structure is often referred to as Deep Boltzmann Mchine (DBM), which stacks several RBMs to build up a hierarchical representation. Without loss of generality, we use a 3-layer architecture to model its connections. The energy function has the following form

$$\begin{aligned} E(\mathbf{v},\mathbf{h}_1,\mathbf{h}_2) = &-\mathbf{a}^\top\mathbf{v} - \mathbf{b}^\top\mathbf{h}_1 - \mathbf{c}^\top\mathbf{h}_2 \\ &- \mathbf{v}^\top\mathbf{W}_1\mathbf{h}_1 - \mathbf{h}_1^\top\mathbf{W}_2\mathbf{h}_2. \end{aligned} \tag{2}$$

- **Shape Boltzmann Machine**. ShapeBM is proposed to deal with the problem when there are only a small amount of training data. Local receptive field is enforced as the block-based connection between the visual layer $\mathbf{v}$ and the first hidden layer $\mathbf{h}_1$. Each patch overlaps with its neighbors to a certain extent and weights are shared between four sets of hidden units and visual nodes. Its energy function is

$$\begin{aligned} E(\mathbf{v},\mathbf{h}_1,\mathbf{h}_2) = &-\mathbf{a}^\top\mathbf{v} - \mathbf{b}^\top\mathbf{h}_1 - \mathbf{c}^\top\mathbf{h}_2 \\ &- \sum_i \mathbf{v}_i^\top\mathbf{W}_1\mathbf{h}_{1i} - \sum_i \mathbf{h}_{1i}^\top\mathbf{W}_2\mathbf{h}_2. \end{aligned} \tag{3}$$

Nevertheless, as far as hand shape images are considered, we notice that the palm pixels are independent of the finger

pixels and the hand shape often mainly differ in finger configurations. Fig. 3 shows the weights learnt by RBM and DBM for hand shape images. We can observe that the palm region has little correlation with the finger region. This suggests that connectivity can be carefully imposed. We introduce the Hand Boltzmann Machine (HBM) to consider the connection between the central palm pixels and the peripheral finger pixels separately, and merge their connection in an upper hidden layer.

III. Hand Boltzmann Machine (HBM)

Let a hand shape image be represented by a binary vector $\mathbf{v}$ with pixel $v_i \in \{0, 1\}$, and $P(\mathbf{v})$ describes how probable $\mathbf{v}$ appears in the hand shape space. As the state of one pixel depends on that of the others, their latent correlations can be characterized as links between the elements $\{v_i\}$ of the image vector $\mathbf{v}$ and the binary nodes $\{h_j\}$ of the hidden variables $\mathbf{h}$. Similar to DBM and ShapeBM, HBM has two layers of latent variables. Moreover, the connectivity between the visual layer and the first layer is manually set according to Fig. 3(c) and 3(d). It consists of two components, $\mathbf{h}_p$ for central palm pixels and $\mathbf{h}_f$ for peripheral finger pixels. Each node of $\mathbf{h}_f$ is connected to all finger pixels in the visual layer, and each node of $\mathbf{h}_p$ to all palm pixels in the visual layer. These two regions of the visual layer overlap with each other by b-pixel width to enforce smoothness of the hand shape. In the second layer, each node is fully connected to all nodes in the first layer, $\mathbf{h}_p$ and $\mathbf{h}_f$. The joint probability of $\mathbf{v}$, $\mathbf{h}_p$, $\mathbf{h}_f$and $\mathbf{h}_2$ is given by

$$P(\mathbf{v}, \mathbf{h}_p, \mathbf{h}_f, \mathbf{h}_2) = \frac{1}{Z} e^{-E(\mathbf{v}, \mathbf{h}_p, \mathbf{h}_f, \mathbf{h}_2)}, \tag{4}$$

where $Z = \sum e^{-E(\mathbf{v}, \mathbf{h}_p, \mathbf{h}_f, \mathbf{h}_2)}$ is a normalization term, and the energy function is

$$\begin{aligned} E(\mathbf{v}, \mathbf{h}_p, \mathbf{h}_f, \mathbf{h}_2) = &- \mathbf{a}^\top \mathbf{v} - \mathbf{b}_p^\top \mathbf{h}_p - \mathbf{b}_f^\top \mathbf{h}_f - \mathbf{c}^\top \mathbf{h}_2 \\ &- \mathbf{v}_p^\top \mathbf{W}_{1p} \mathbf{h}_p - \mathbf{v}_f^\top \mathbf{W}_{1f} \mathbf{h}_f \\ &- \mathbf{h}_p^\top \mathbf{W}_{2p} \mathbf{h}_2 - \mathbf{h}_f^\top \mathbf{W}_{2f} \mathbf{h}_2. \end{aligned} \tag{5}$$

By summing over all possible configurations of $\mathbf{h}_p$, $\mathbf{h}_f$ and $\mathbf{h}_2$, we obtain final marginal distribution of the hand shape image $\mathbf{v}$ by

$$P(\mathbf{v}) = \frac{1}{Z} \sum_{\mathbf{h}_p, \mathbf{h}_f, \mathbf{h}_2} e^{-E(\mathbf{v}, \mathbf{h}_p, \mathbf{h}_f, \mathbf{h}_2)}. \tag{6}$$

The first layer encodes different finger shape variances and palm variances, which can be seen as the local properties of hands. The second layer imposes global constrains, such as global shape smoothness, finger configuration, *etc*. It holds the higher level information of different hand shapes.

A. Learning the Model

We follow the procedure of training a DBM [13] to maximize the log-likelihood $\log P(\mathbf{v}; \Theta)$ of the observed hand images with respect to the parameters $\Theta = \{\mathbf{a}, \mathbf{b}_p, \mathbf{b}_f, \mathbf{c}, \mathbf{W}_{1p}, \mathbf{W}_{1f}, \mathbf{W}_{2p}, \mathbf{W}_{2f}\}$ for a Hand Boltzmann Machine model.

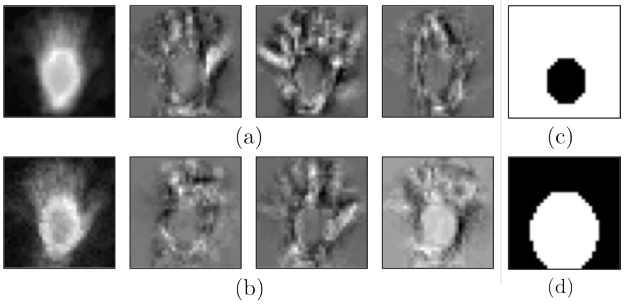

Fig. 3. (a) Weights $\mathbf{W}_{1i}$ for visual nodes of 4 hidden units of a trained RBM. (b) Weights of a trained DBM. Note that the palm pixels in the center of the image have strong correlation with each other, although weights vary a lot for different hidden units. (c) The active finger pixels (in white) connected to $\mathbf{h}_f$. (d) The active palm pixels (in white) connected to $\mathbf{h}_p$.

There are two phases to learn a HBM model. In the first phase, a greedy, layer-by-layer *pre-training* algorithm is used to learn one layer at a time. It starts from the bottom visual layer $\mathbf{v}$, and learns $\mathbf{W}_{1p}$ for the palm RBM and $\mathbf{W}_{1f}$ for finger RBM simultaneously. This is done by Persistent CD algorithm [14]. The connectivity of palm and finger RBMs is defined manually by considering the central-peripheral property in Fig. 3. After both RBMs are trained, the values of nodes in $\mathbf{h}_p$ and $\mathbf{h}_f$ are inferred according to Eq. 7 for each training instance to initialize the upper RBM. In the second phase, three RBMs of two layers are composed to create a HBM model. The parameters obtained in the *pre-training* are used to initialize the HBM model which is *fine-tuned* in the second phase still using contrastive divergence algorithm. In this way, global and local variances of the hand shape images spread out through the bottom-up input and top-down feedback process.

IV. Training Hand Shape Classifier from Sketch Input

After the hand shape model is obtained, it is ready to use sketch to generate samples for training a hand shape classifier.

A. Sketch Input as Hard Constraint in Sampling

Interestingly, each node v_i or h_j in the same layer is conditionally independent of other nodes once the opposite layer is observed, and they are conditionally distributed as

$$\begin{aligned} &P(v_j = 1 | \mathbf{h}_p, \mathbf{h}_f) = \sigma(a^j + \textstyle\sum_{i \in \mathcal{P}} W_{1p}^{ji} h_p^i + \sum_{i \in \mathcal{F}} W_{1f}^{ji} h_f^i) \\ &P(h_p^j = 1 | \mathbf{v}, \mathbf{h}_2) = \sigma(b_p^j + \textstyle\sum_{i \in \mathcal{P}} W_{1p}^{ij} v_i + \sum_i W_{2p}^{ji} h_2^i) \\ &P(h_f^j = 1 | \mathbf{v}, \mathbf{h}_2) = \sigma(b_f^j + \textstyle\sum_{i \in \mathcal{F}} W_{1f}^{ij} v_i + \sum_i W_{2f}^{ji} h_2^i) \\ &P(h_2^j = 1 | \mathbf{h}_p, \mathbf{h}_f) = \sigma(c^j + \textstyle\sum_i W_{2p}^{ij} h_p^i) + \sum_i W_{2f}^{ij} h_f^i) \end{aligned}, \tag{7}$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$ is the sigmoid function, and $\mathcal{P}$ and $\mathcal{F}$ denote the indices of palm and finger pixels. This allows for the layerwise Gibbs sampling as illustrated in Fig. 4. $\mathbf{v}$, $\mathbf{h}_p$, $\mathbf{h}_f$ and $\mathbf{h}_2$ are iteratively sampled in a layer by layer fashion according to Eq. 7

Given a sketch image $\mathbf{I}_s$, each of its pixels p_i has one of the three possible values: $\{hand, background, unknown\}$. In the

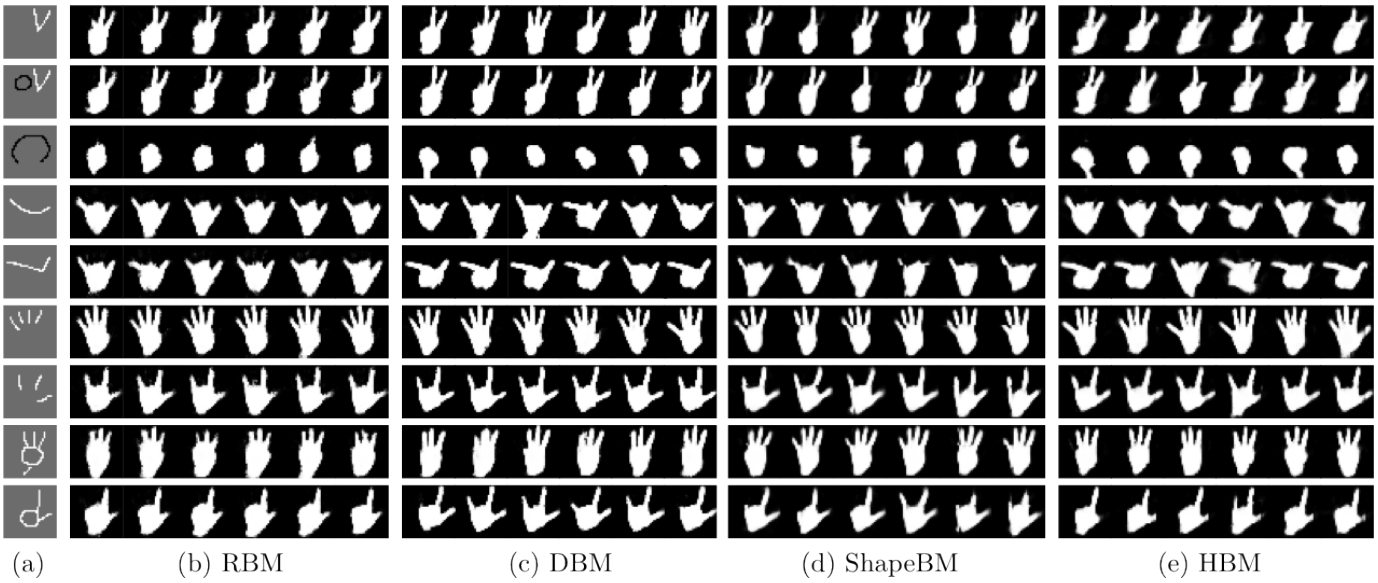

Fig. 5. Some examples after 10-step Gibbs sampling. (a) Sketches drawn by the users. Black strokes indicate *background* pixels, white strokes indicate *hand* pixels and grey color indicates *unknown* pixels. Each sketch is first scaled to 32×32 pixels and then used as input to different models. (b) Samples generated by RBM. (c) Samples generated by DBM. (d) Samples generated by ShapeBM. (e) Samples generated by our HBM.

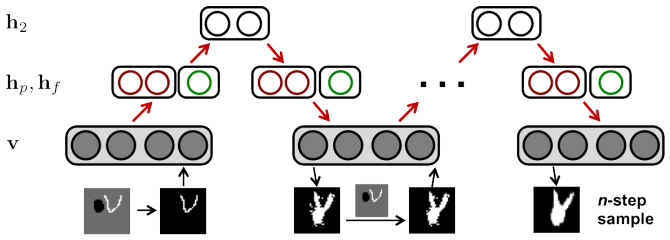

Fig. 4. Sampling procedure with a sketch. In the first iteration, only *hand* pixels are considered as 1 for sampling and the rest as 0. In the following iterations, the sampled vector is constrained by the sketch, whose *hand* pixels are set to be 1 and *background* pixels to be 0, before being used as the input for the next iteration.

first step of sampling, all *background* pixels and *unknown* pixels are set to 0, and the rest *hand* pixels are set to 1. In the following steps, only *hand* and *background* pixels are constrained to be 1 and 0 respectively. After n-step Gibbs sampling, we use the sample $\hat{\mathbf{v}}$ as the approximated hand shape and its hidden state $\hat{\mathbf{h}}_2$ to train the classifier.

B. Building up a Classifier from Samples

For each sketch, we run sampling m times to collect both visual sample vectors $\{\hat{\mathbf{v}}^{(m)}\}$ and hidden sample vectors $\{\hat{\mathbf{h}}_2^{(m)}\}$. They are further used for training either linear or non-linear classifier. In our experiment, we test two settings:

- A hand shape classifier of visual input is trained from the inferred hand shape samples generated from sketches. This is referred to as *visual classifier*.
- A hand shape classifier of hidden state variables is trained from the hidden vectors corresponding to the aforementioned hand shape samples. This is referred to as *hidden classifier*.

C. Testing

During testing phase, a normal hand image is preprocessed by the model. This process is necessary to complete the partially observed data and remove noisy pixels. After n-step sampling, the sample of visual layer is used for *visual classifier*, while that of hidden layer is used for *hidden classifier*.

V. Experimental Results

In our experiment, hand images were obtained from Intel Gesture Camera [3], and thresholded to produce binary segmentations and normalized to 32×32 pixels. We recorded images of different hand configurations of 6 subjects. Each subject performed 15 hand postures and some free hand movement, *e.g.*, jiggle. We trained four models, namely RBM, DBM, ShapeBM and HBM, with 70% of these images and used the rest images for testing. There were 500 hidden nodes in the RBM model and they were trained for 500 epochs. The DBM model consisted of 1000 and 200 nodes for $\mathbf{h}_1$ and $\mathbf{h}_2$. The first layer was pre-trained for 500 epochs and the second layer for 500 epochs. After pre-training, the whole model was fine-tuned for 300 epochs. The ShapeBM model had 4 patches overlapped by 4-pixel width and for each patch there were 500 hidden nodes in the first layer. These 2000 hidden nodes were fully connected to 200 hidden ones in the upper layer. The first and second layer were pre-trained for 500 and 500 epochs respectively before the model was jointly trained for another 300 epochs. Finally for HBM, the central palm region and the peripheral finger region overlapped with each other by 4-pixel width and there were 500 and 1000 nodes for the palm and finger region respectively in the first layer, and 200 nodes in the second layer. Similar to the DBM and the ShapeBM models, the HBM model was trained for 500 and

500 epochs for the first and second layer, and fine-tuned for 300 epochs in the joint training. We evaluated our method in three aspects, namely its capability to handle different sketch input, its classification accuracy, and its capability to handle occlusions.

A. Samples from different sketches

We developed a simple GUI that allows users to draw sketches for each hand shape intuitively. Fig. 5 shows the samples generated from the sketches collected from different users. The shapes were sampled after 10 iterations. The first three rows show that ambiguity can be eliminated by introducing the $background$-pixel constraint, the next 5 rows list some abstract sketches, and in the last row is a sketch indicating hand posture which is unseen in the training set. Follow the evaluation of [7], we investigate our results from two aspects, *realism* and *generalisation*.

1) Realism: All four models can generate realistic samples. RBM cannot change the shape too much due to its shallow structure. DBM is the most connected model and we can see great variation of the samples. ShapeBM puts more emphasis on local block area, so the variation is more within a local patch, such as the abnormal sample in the forth row. HBM separates the connections between palm pixels and finger pixels, so we can observe that the shape of palm varies regardless of the variation of fingers. For instance, all fingers of the samples look similar but all palms differ a lot in size in the eighth row. This shows that the configuration of fingers are independent of palms. Although different style of sketches are used here, we can see that they all generate meaningful hand shape images.

2) Generalization: We also investigated whether these four models can generate novel samples that are not seen in the training set. In Fig. 5, the last row shows a novel hand posture which is not in the training set. RBM generated similar hand shape images, but they look the same. DBM generated samples that are similar to the ones in the dataset. ShapeBM generated samples with local changes, but some of the samples are not realistic. HBM generated shapes that do not only follow the guidance but also look realistic. The various finger configurations can be further eliminated by applying more constraints.

B. Classification results

The subject was asked to draw 5 sketches for each class respectively to describe the 15 classes as shown in Fig. 6. 100 samples were generated for each sketch. We trained an SVM classifier using `libsvm` [15] and a generalized linear model using `glmnet` [16] on these images and 200 hand shape images of 6 different subjects were used for testing. We repeated this process 10 times using our $Matlab$ implementation on a standard PC with a 2.5 GHz dual-core CPU.

Table I shows the results of *visual* and *hidden* classifiers for different models, trained by different number of sketches, and we have the following four observations.

		1 Sketch		5 Sketches	
		Linear	SVM	Linear	SVM
RBM	$\mathbf{v}$	32%	33.4%	59.7%	73%
	$\mathbf{h}_2$	36%	38.8%	63.5%	79.2%
DBM	$\mathbf{v}$	42.7%	43.2%	76.5%	82.73%
	$\mathbf{h}_2$	53.9%	55.1%	69.2%	78.2%
ShapeBM	$\mathbf{v}$	41.8%	45.8%	75.1%	78%
	$\mathbf{h}_2$	53.9%	56.9%	78.1%	81.2%
HBM	$\mathbf{v}$	45.2%	46.4%	76.9%	82.5%
	$\mathbf{h}_2$	55.3%	58%	83.5%	85.8%

TABLE I
CLASSIFICATION ACCURACY

1) $\mathbf{v}$ vs. $\mathbf{h}_2$: The classification results of hidden layer are better than those of visual layer for all models except for DBM. It suggests that the hidden layer handles more shape variance as it corresponds to higher level shape properties. Moreover, we can use hidden code, whose length is 19.5% of the visual one, as a compact representation of a shape image.

2) Number of sketches: We generated the samples from 1 to 5 sketches of different styles to illustrate different hand postures. We found that the result improves with increasing number of sketches. In other words, during the classifier building phase, a user may draw additional sketch similar to mistakenly classified result iteratively to "complete" the classifier.

3) Linear vs. non-linear: `svm` performed better than `glmnet` in most cases. It suggests that the high dimensional vector is still non-linear no matter it is from the visual or hidden layer.

4) HBM vs. the rest: RBM cannot deal with various shape changes when the number of training sketches is small because it only consider the local shape variance due to its shallow structure. On the contrary, the samples vary a lot for DBM, which in turn cause some misclassification when building the classifier. In ShapeBM, block-based constraints are applied so that it performs better than DBM as the samples for training are less ambiguous. HBM applies the hand-specific constraint so that the variance of palm and fingers are separated. We found that classification benefit from such constraints so the results are the best among these models.

More specifically, we investigated the confusion matrix of classification results for HBM trained with 5 sketches (See Fig. 6). We found that the visual confusion and hidden confusion are quite similar.

In addition, the average time of processing one image during testing is shown in Table. II. An image is preprocessed after 10-step sampling and *Hidden* classifiers by SVM were profiled here. The time cost for classification are similar for the four models, as the lengths of the input vectors for the SVM classifiers are similar. However, it takes much more time for deep models to pre-process the input image than RBM. As the size of parameters in HBM model lies in between that of the DBM model and the ShapeBM model, the time cost also

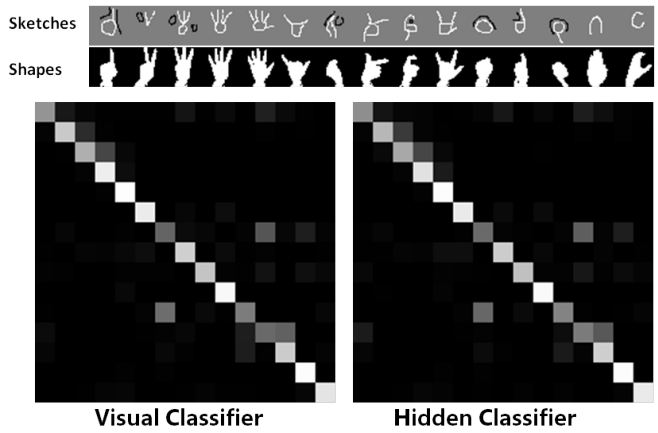

Fig. 6. Confusion matrices of 15 classes listed in the top rows. The sketches used for training is listed on the first row and the representative samples on the second row.

lies in the middle. Note that the total time cost for the HBM model is 89ms, which means that it is possible for real-time applications after optimizing the code.

	RBM	DBM	ShapeBM	HBM
Preprocessing (ms)	5	125	93	86
Classification (ms)	2	4	3	3
Total	7	129	96	89

TABLE II
TIME COST DURING THE TESTING PHASE

C. Capability to Handle Occlusions

In real world applications, occlusions are very common for hand shape recognition systems. For example, a hand may hold some object, *e.g.*, a pencil, an orange, a cup, *etc*. We obtained a partial observation of the hand after thresholding on depth and detecting skin color. In order to apply our method, the complete hand shape image can be estimated by n-step Gibbs sampling from the trained HBM (see Fig. 7). The generated hand shape estimate is quite realistic even when there is large occlusion. Therefore, preprocessing is necessary here to generate hidden input and also complete the partial observation, which makes it more applicable in real world scenario.

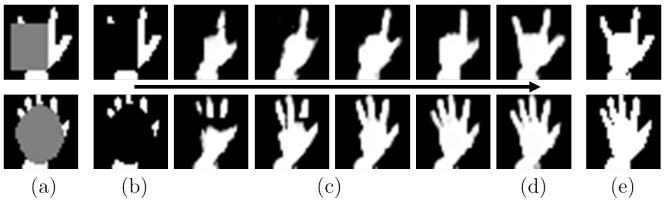

Fig. 7. (a) Occluded images. (b) Initial inputs. (c) Interim results in sampling. (d) Estimated complete images after sampling. (e) Original image examples.

VI. CONCLUSIONS

In this paper, we propose a method that allows a user to define a hand shape recognizer by sketching the hand shape intuitively. The core idea of our method is to learn a probabilistic model, the Hand Boltzmann Machine, by unsupervised learning from available hand shape images. Both sketches and hand images can be viewed as the guidance for sampling similar hand shape configurations from the probabilistic shape model. This is similar to the view of poselet [17] on the appearance space and its latent configuration space. It can also be viewed as retrieving the samples similar to the sketch from a probabilistic distribution instead of a database. Since our method also benefits from the generative model to complete partially observed data, it has great potential for the scenario where there are intensive hand-object interactions in real world applications. Note that, however, our method is based on the appearance of hand shape, it thus cannot deal with large variance of different hand poses simply by one single sketch.

REFERENCES

[1] Juan Pablo Wachs, Mathias Kölsch, Helman Stern, and Yael Edan, "Vision-based hand-gesture applications," *Communications of the ACM*, vol. 54, no. 2, pp. 60, Feb. 2011.
[2] "Kinect for Xbox 360," .
[3] "The CREATIVE Interactive Gesture Camera," .
[4] Zhou Ren, Junsong Yuan, and Zhengyou Zhang, "Robust hand gesture recognition based on finger-earth mover's distance with a commodity depth camera," *Proc. 19th ACM International Conference on Multimedia*, pp. 1093–1096, 2011.
[5] Nicolas Pugeault and Richard Bowden, "Spelling It Out : Real-Time ASL Fingerspelling Recognition," in *Proc. ICCV 2011 Workshop on Consumer Depth Cameras for Computer Vision*, 2011.
[6] Marc'Aurelio Ranzato, Joshua Susskind, Volodymyr Mnih, and Geoffrey Hinton, "On deep generative models with applications to recognition," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*. June 2011, pp. 2857–2864, Ieee.
[7] SMA Eslami and Nicolas Heess, "The shape boltzmann machine: a strong model of object shape," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 406 –413.
[8] Hai Wang, Changhu Wang, Yang Cao, Zhiwei Li, and Lei Liqing Zhang, "MindFinder: interactive sketch-based image search on millions of images," in *Proc. 18th ACM International Conference on Multimedia*, New York, New York, USA, Oct. 2010, pp. 1605–1608, ACM Press.
[9] Dariu M Gavrila, "A Bayesian, exemplar-based approach to hierarchical shape matching.," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 8, pp. 1408–21, Aug. 2007.
[10] M.P. Kumar, P.H.S. Ton, and A. Zisserman, "OBJ CUT," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2005, pp. 18–25.
[11] E. Borenstein, E. Sharon, and S. Ullman, "Combining Top-Down and Bottom-Up Segmentation," in *Proc. CVPR 2004 Workshop on Perceptual Organization in Computer Vision*. 2004, pp. 1–8, Ieee.
[12] Yuri Y Boykov and M.-P. Jolly, "Interactive Graph Cuts for Optimal Boundary & Region Segmentation of Objects in N-D Images," in *Proc. 8th IEEE International Conference on Computer Vision*, 2001, number July, pp. 105–112.
[13] Ruslan Salakhutdinov and Geoffrey Hinton, "Deep Boltzmann Machines," in *Proc. 12th International Conference on Artificial Intelligence and Statistics*, 2009, number 2, pp. 1–8.
[14] GE E Hinton, Simon Osindero, and YW W Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.
[15] Chih-Chung Chang and Chih-Jen Lin, "LIBSVM: A library for support vector machines," *ACM Trans. on Intelligent Systems and Technology*, vol. 2, no. 3, pp. 1–27, Apr. 2011.
[16] Jerome Friedman, Trevor Hastie, and Rob Tibshirani, "Regularization Paths for Generalized Linear Models via Coordinate Descent.," *Journal of statistical software*, vol. 33, no. 1, pp. 1–22, Jan. 2010.

[17] Lubomir Bourdev and Jitendra Malik, "Poselets: Body part detectors trained using 3D human pose annotations," in *Proc. 12th IEEE International Conference on Computer Vision*, 2009, pp. 1365–1372.