# Learning Fingerprint Orientation Fields Using Continuous Restricted Boltzmann Machines

Mihir Sahasrabudhe and Anoop M. Namboodiri
*Centre for Visual Information Technology, IIIT Hyderabad, India*
{mihir.s@research., anoop@}iiit.ac.in

***Abstract*—We aim to learn local orientation field patterns in fingerprints and correct distorted field patterns in noisy fingerprint images. This is formulated as a learning problem and achieved using two continuous restricted Boltzmann machines. The learnt orientation fields are then used in conjunction with traditional Gabor based algorithms for fingerprint enhancement. Orientation fields extracted by gradient-based methods are local, and do not consider neighboring orientations. If some amount of noise is present in a fingerprint, then these methods perform poorly when enhancing the image, affecting fingerprint matching. This paper presents a method to correct the resulting noisy regions over patches of the fingerprint by training two continuous restricted Boltzmann machines. The continuous RBMs are trained with clean fingerprint images and applied to overlapping patches of the input fingerprint. Experimental results show that one can successfully restore patches of noisy fingerprint images.**

***Keywords*-Fingerprint Enhancement, Biometrics, Continuous Restricted Boltzmann Machines, Gabor Filters, Machine Learning**

## I. Introduction

Enhancement of fingerprint images plays an important part in fingerprint based authentication systems. This step plays a vital role as the noise in fingerprint images can be removed before feature extraction or matching, reducing the probability of a false match. Noise in fingerprints can appear because of several reasons including, but not limited to, sensor noise, finger dampness or dryness, bruises and cuts in fingers, and non-uniform finger pressure [1]. Several approaches to fingerprint enhancement have been proposed in the past. Some of these approaches, like median filtering, contrast limited adaptive histogram equalization, and Wiener filtering, try to model the noise in the fingerprint image and remove it. However, they are not very successful in modeling the kind of noise present in the fingerprint image for it can be a mixture of several kinds of noise.

Neural networks started gaining popularity again in the mid-2000s as new methods to train them, much better than existing ones, surfaced [8]. The use of many-layered neural networks to learn patterns gave rise to the term "deep learning". One model for this is the deep belief net, built by stacking several restricted Boltzmann machines (RBMs) over each other. The RBMs, as independent models, have been used as generative models for several years [10], [12], [13], [11] with the problems addressed by them [9] spanning from bag-of-words for document representation [13] to user ratings of movies [12] and modeling video [10] and speech [11]. Traditionally, RBMs work only on binary inputs. Chen and Murray [2] proposed a model, the continuous RBM or CRBM, which is a variant of the RBM, and can work on continuous inputs. We shall use this model in this work to encode and learn orientation fields of fingerprint images.

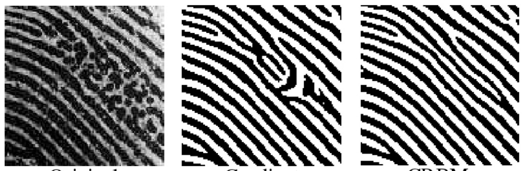

Figure 1. Performance of the proposed algorithm with respect to local correction of orientation field compared to gradient-only orientation field extraction

## II. Related Work

Several techniques currently used in fingerprint enhancement involve contextual filtering - a primary filter used being the Gabor filter. The use of Gabor filter for fingerprint enhancement was proposed by Hong, *et al.* [3]. Parameters for the filters used on the image are determined by local ridge orientation, local ridge frequency and reliability of ridge orientation. Unlike these, methods that operate in the frequency domain [4] have also been attempted. The algorithm proposed Hong *et al.* involves a bank of Gabor filters that have optimal joint resolution in both spatial and frequency domains. This method has undergone several changes ([5], [6]).

In this work, we attempt to learn various types of local orientation field patterns observed in fingerprints by training a neural network instead of specifying them in the model. We used two continuous restricted Boltzmann machines for this task. They were trained in an unsupervised manner initially, and then a pass of backpropagation learning was introduced to further correct the weight matrices. As the network is trained on clean fingerprint images, the application of the network to noisy orientation fields results in them being approximated by a cleaner one. Figure 1 shows a comparison between results of gradient-based methods and the proposed approach.

## III. Proposed Approach

The proposed approach is divided into three stages: generating orientation fields of query images using a gradient-based method; training two continuous restricted Boltzmann machines (CRBMs) on a database of noiseless orientation field images; and feeding the orientation fields of query images to these RBMs, and using the outputs for Gabor filtering. We'll describe each stage here.

### *A. Gradient-based approach to estimate orientation field*

We use the algorithm given by Hong *et al.* in [3] for an initial estimate of the orientation field. This algorithm performs very well in noiseless images, but doesn't give robust estimates of the orientation field for noisy images as it considers a small region during estimation. The process of obtaining this orientation field is summarized here:

1) Divide the given image into blocks of sizes $W \times W$. Consider a block centered at a pixel $(i, j)$. Compute the horizontal and vertical gradients using Sobel operators to obtain $\partial_x(i, j)$ and $\partial_y(i, j)$, respectively.
2) Obtain an estimate of the orientation field, $\theta(i, j)$ using the following equations:

$$\mathcal{V}_x(i,j) = \sum_{u=i-\frac{W}{2}}^{i+\frac{W}{2}} \sum_{v=j-\frac{W}{2}}^{j+\frac{W}{2}} 2\partial_x(u,v)\partial_y(u,v) \tag{1}$$

$$\mathcal{V}_x(i,j) = \sum_{u=i-\frac{W}{2}}^{i+\frac{W}{2}} \sum_{v=j-\frac{W}{2}}^{j+\frac{W}{2}} \partial_x^2(u,v)\partial_y^2(u,v) \tag{2}$$

$$\theta(i,j) = \frac{1}{2}\arctan\frac{\mathcal{V}_y(i,j)}{\mathcal{V}_x(i,j)} \tag{3}$$

3) Convert the orientation field into a continuous vector field:

$$\Phi_x(i,j) = \cos(2\theta(i,j)) \tag{4}$$

$$\Phi_y(i,j) = \sin(2\theta(i,j)) \tag{5}$$

Perform Gaussian smoothing to get $\Phi'_x$ and $\Phi'_y$:

$$\Phi'_x(i,j) = \sum_{u=\frac{-w_\phi}{2}}^{\frac{w_\phi}{2}} \sum_{v=\frac{-w_\phi}{2}}^{\frac{w_\phi}{2}} G(u,v)\Phi_x(i-uw, j-uw) \tag{6}$$

$$\Phi'_y(i,j) = \sum_{u=\frac{-w_\phi}{2}}^{\frac{w_\phi}{2}} \sum_{v=\frac{-w_\phi}{2}}^{\frac{w_\phi}{2}} G(u,v)\Phi_y(i-uw, j-uw) \tag{7}$$

where $G$ is a Gaussian low-pass filter of size $w_\phi \times w_\phi$. The orientation field is now obtained as:

$$O(i,j) = \frac{1}{2}\arctan\frac{\Phi'_y(i,j)}{\Phi'_x(i,j)} \tag{8}$$

The orientation field calculated using this approach is used as input in the next step. It is important to note that the orientation field values generated by this algorithm lie in the range $[0, \pi]$, as an orientation of $\alpha$ and $\pi + \alpha$ is essentially the same.

### *B. Training the CRBMs*

The focus of our approach is training of the continuous restricted Boltzmann machines. The RBMs are trained using noiseless orientation fields so that they learn fingerprint ridge patterns. The training images used in this step were obtained by orientation field estimation of fingerprint patches taken from enhanced images. It was ensured that these enhanced images did not have noise, so that the training samples were noiseless.

**The Model:** The classical RBM takes only binary inputs, so we used a variant of the same - the continuous restricted Boltzmann machine (CRBM) [2] - as our model. The CRBM inputs are normalized to $[0, 1]$. We train two CRBMs, hence the significance of $0$ and $1$ for both of them is different. To ensure that we get continuity in our output, we don't use the orientation value directly for training. To emphasize this, consider the orientation field values near the apex of an upward-curved ridge. Approaching the apex from left, the orientation tends to $\pi$, whereas approaching it from the right, it tends to $0$. It is due to this inconsistency that we do not use orientation values for training. Training a CRBM using the orientation field, $\theta$, gives rise to anomalies in the resulting image when the orientation has to jump from $\pi$ to $0$. This gives rise to inconsistent outputs of the CRBM, wherein it doesn't jump directly from $\pi$ to $0$. To counter this, we break down $\theta$ into two components, by applying the functions $s(\theta)$ and $c(\theta)$.

$$s(x) = \begin{cases} \frac{4}{\pi}x & \text{for } 0 \le x < \frac{\pi}{4}; \\ -\frac{4}{\pi}x + 2 & \text{for } \frac{\pi}{4} \le x < \frac{3\pi}{4}; \\ \frac{4}{\pi}x - 4 & \text{for } \frac{3\pi}{4} \le x \le \pi \end{cases} \tag{9}$$

$$c(x) = \begin{cases} -\frac{4}{\pi}x + 1 & \text{for } 0 \le x < \frac{\pi}{2}; \\ \frac{4}{\pi}x - 3 & \text{for } \frac{\pi}{2} \le x \le \pi \end{cases} \tag{10}$$

**Parameters and training:** As described in [2], a CRBM is different from the traditional RBM. The CRBM has two layers - visible (or input) and hidden - wherein every neuron in the visible layer is connected to every neuron in the hidden layer. These are undirected, weighted edges, hence the hidden layer can feed inputs to the visible layer as well. A neuron in either layer emulates a sigmoid function. This sigmoid is applied on the total input received at that neuron plus a zero-mean Gaussian noise. Let $j$ be a neuron in the hidden layer, and let $\mathbf{v} = [v_1, v_2, \ldots, v_V]^T$ be the vector denoting the values at neurons at the visible layer. Let $\mathbf{w}_j = [w_{j1}, w_{j2}, \ldots, w_{jV}]^T$ be the set of weights corresponding to the $j$-th neuron in the hidden layer. An additional parameter is introduced for every neuron in a layer for the CRBM. This parameter, called the "noise-control" parameter and denoted by $a_j$, controls the nature of the neuron's stochastic behavior. The output of the $j$-th neuron in the hidden layer is then given by

$$s_j = \phi(\theta_j) = \frac{1}{1+\exp(-\theta_j)}; \text{where } \theta_j = a_j x_j \tag{11}$$

and $x_j$ denotes the total input at neuron $j$:

$$x_j = \sum_{i=1}^{V} w_{ji} v_i + b_j + \sigma N(0,1) \tag{12}$$

$\sigma$ and $N(0,1)$ make up the noise component, where $\sigma$ is a constant for the whole network, and $N(0,1)$ is a Gaussian random variable with zero mean and unit variance. Both $\mathbf{w}_j$ and $a_j$ are updated for every neuron $j$ after each epoch. The quantity $b_j$ is the bias added to every unit. The update rules are given by

$$\Delta w_{ji} = \eta_w \left(\langle v_i s_j \rangle - \langle \hat{v}_i \hat{s}_j \rangle\right) \tag{13}$$

$$\Delta a_j = \frac{\eta_a}{a_j^2} \left(\langle s_j^2 \rangle - \langle \hat{s}_j^2 \rangle\right) \tag{14}$$

where $\hat{v}_i$, $\hat{s}_j$ denote the one-step sampled state of neurons $i$ and $j$, respectively; $\eta_w$ and $\eta_a$ are learning rates; and $\langle \cdot \rangle$ denotes the expected value over all training samples. The learning rates for the c-RBM used in training were $\eta_w = 5$ and $\eta_a = 0.5$, while those used for the s-RBM were $\eta_w = 4$ and $\eta_a = 0.5$. The set of weights used to calculate activations of hidden neurons from the visible neurons is the transpose of the set of weights used to calculate the activations of the visible neurons from the hidden neurons.

In this work, we focused on training with fingerprint patches of sizes 60 pixels $\times$ 60 pixels. However, exploiting the similarity in orientation field observed in pixels which are neighbors, we used only 100 neurons in the visible layer of CRBMs. This was done by first resizing the training patches to $10 \times 10$, and then assigning each pixel in the resulting patch a unique neuron in the visible layer. In the hidden layer, we used 90 neurons. Training was performed using 4580 training samples, with 25000 epochs per CRBM.

The CRBMs were trained in an unsupervised manner according to these training rules, and were then subject to a pass of backpropagation learning to fine tune the weights. Before this pass, the CRBMs are "opened" up to get an output layer independent of an input layer. In this process, the symmetric nature of weights in the pretraining phase is removed, and both set of weights become independent of one another. The backpropagation algorithm for the CRBM is different from the one for an RBM because of parameters not included in the RBM. The updates for backpropagation learning hence needed to be deduced. These have been laid out here. Call the number of layers in the opened up network, $L$ (starting from 0). Here, $L = 2$ for the CRBMs. Let $y_i$ denote the expected/ideal value of the output at neuron $i$ in the output layer. Let $X_i^l$ denote the output of the $i$-th neuron

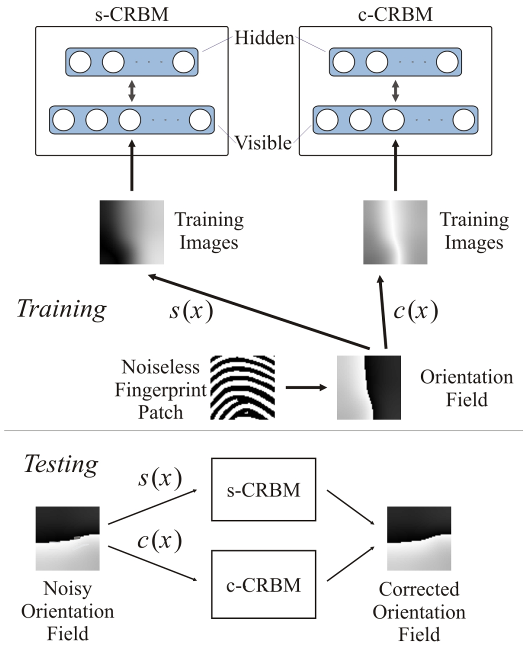

Figure 2. The model, training and testing processes. *top (training)*: A representation of the continuous RBM used in this work. The first two steps of the proposed approach are summarized. *bottom (testing)*: A representation of the third step in the proposed approach.

in layer $l$, $w_{ij}^l$ denote the weight between $i$-th neuron in layer $l$ and $j$-th neuron in layer $l-1$; $n_l$ denote the number of neurons in layer $l$; and $a_i^l$ denote the noise-control parameter at the $i$-th neuron in layer $l$. The backward pass equation for layer $L$ is as follows:

$$\delta_i^L = (y_i - X_i^L) \cdot a_i^L \cdot \phi'(\theta_i^L) \tag{15}$$

where $\phi'(x) = \phi(x)(1-\phi(x))$ is the derivative of the sigmoid function, and $\theta_i^L$ is the product of the noise-control parameter and the total input at neuron $i$ given by $\theta_i^L = a_i^L \left( \sum_{j=1}^{n_{L-1}} w_{ij}^L X_j^{L-1} + b_j + \sigma N(0,1) \right)$. Backward pass in the remaining layers is given by:

$$\delta_i^l = \phi'(\theta_i^l) \cdot a_i^l \cdot \sum_{t=1}^{n_{l+1}} \delta_t^{l+1} w_{ti}^{l+1}; \text{where } 0 < l < L \tag{16}$$

Equations 15 and 16 lead to the following update rules for $\mathbf{w}_j$ and $a_j$:

$$\Delta w_{ij}^l = \nu_w \cdot \delta_i^l \cdot X_j^{l-1} \tag{17}$$

$$\Delta a_i^l = \nu_a \cdot \frac{\delta_i^l}{a_i^l} \cdot x_i^l = \nu_a \cdot \frac{\delta_i^l}{(a_i^l)^2} \cdot \theta_i^l \tag{18}$$

where $x_i^l = \left( \sum_{j=1}^{n_{l-1}} w_{ij}^l X_j^{l-1} + b_i + \sigma N(0,1) \right)$ is the total input received at the $i$-th neuron in layer $l$, and $\nu_w$ and $\nu_a$ are learning rates for the weights and the noise-control parameter, respectively. In our experiments, we set the learning rates for the weights and the noise control parameter to be equal. The values used were $\nu_w = \nu_a = 1$ for the c-RBM, and $\nu_w = \nu_a = 0.9$ for the s-RBM.

Both CRBMs are trained with backpropagation learning according to these update rules for 50000 epochs.

### C. *Gabor filtering using outputs of the CRBM*

Once the CRBMs have been trained, they are fed inputs from a database. We have used the FVC 2002 Db3_a (http://bias.csr.unibo.it/fvc2002/databases.asp) database in this paper. Before feeding these images to the CRBMs, their orientation fields are calculated using the algorithm mentioned in section III-A. Then two images are generated for every image in the database by applying the functions $s(x)$ and $c(x)$ on the orientation fields. These images are then fed to the respective CRBMs and their outputs recorded. Now, using the two output images for every image in the database, we can generate the orientation field for that image. Let $cIm$ and $sIm$ denote, respectively, the outputs for the c-CRBM and the s-CRBM for an image $Im$. We'll use them to get the required orientation field. Let $oIm$ be an image, the same size as $cIm$ and $sIm$. For all pixels in $sIm$ whose value is greater than or equal to $0.5$, let a value of $\frac{\pi}{4}(1-c)$ be assigned to the corresponding pixels in $oIm$, where $c$ denotes the values at corresponding pixels in $cIm$. We do the same for pixels with value less than $0.5$, except that a value of $\frac{\pi}{4}(3+c)$ is assigned instead. The matrix $oIm$ now has the required orientation field.

The above sequence of steps generates an orientation field from the s- and c-CRBM outputs, which can directly be used for Gabor filtering. Instead of orientation generated using our gradient-based method (section III-A), we use the orientation field generated by the CRBMs. The frequency image can directly be obtained from the original image. The frequency and orientation images together dictate the natures of the Gabor filters, which are then used appropriately to filter and enhance images from the database.

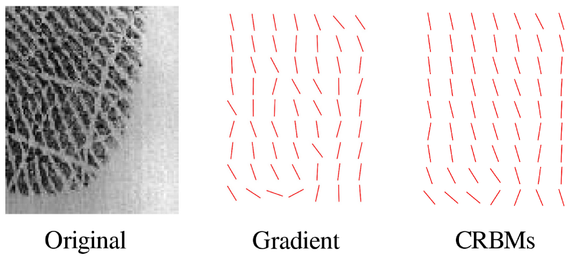

Figure 3. Correction performed in the orientation field by the Continuous RBMs. Orientation fields in the center of the fingerprint patch were distorted by creases. These have been corrected by the CRBMs.

## IV. Experimental Results

We show, on the basis of results of experiments conducted using the above set-up, that this model can correct orientation fields in fingerprints where noise is present in small-sized, local regions.

### A. *Experimental Setup*

The Continuous RBM required for this task was programmed in Python 2.6.5 [GCC 4.4.3]. The program was setup to work on an Nvidia CUDA GeForce GTX 580 graphics processing unit with 512 cores and 1 GB of memory. The cudamat [7] library for python was used to call GPU operations. Unsupervised pre-training for 25000 epochs took 88 minutes per CRBM and backpropagation training for 50000 epochs took 83 minutes. It took an additional 6 minutes for the CRBMs to work on the complete FVC2002 Db3_a database. This gives, approximately, an additional 0.45 seconds per image to the traditional Gabor enhancement.

### B. *Qualitative Analysis*

We perform a qualitative analysis with the gradient-based orientation field enhancement algorithm. We used the code provided by Peter Kovesi [14] to test the gradient-based algorithm against ours. We have tested our algorithm on all fingerprints from the Db3_a database of the FVC 2002 set of databases. It was observed that this algorithm improves on this orientation field generated by the gradient-based algorithm in images where noise is present in small regions and which causes small, local discontinuities in the ridge structure. Further, it was observed that regions where the gradient method performed well weren't distorted by the outputs of the CRBMs. The proposed approach also performs better than gradient-only coupled with Gabor in removing creases from fingerprints. Figure 4 illustrates an example.

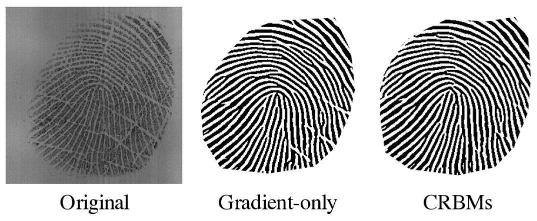

Figure 4. A comparison between Gradient-only orientation field estimation and the proposed approach. The proposed approach is more successful in removing creases, and doesn't distort the rest of the orientation field estimated by the gradient-based method.

### C. *Quantitative Analysis*

Quantitative analysis was performed on this algorithm by plotting the false accept and genuine accept rates output by the NIST matcher, bozorth3 [15], when fingerprints enhanced with the proposed approach were given as input. For minutiae extraction, we used the software FingerJetFXOSE, developed by Digital Persona Inc. For comparison, we have also shown the plot of false accept and genuine accept rates corresponding to the gradient-based Gabor enhancement of fingerprints and STFT-analysis. Figure 5 displays these ROC curves. The database of fingerprints used was again Db3_a of the FVC 2002 set of databases. ROC curves have been plotted after performing 2800 genuine comparisons, and 4950 impostor comparisons.

It can be seen from the ROC cruves that the proposed algorithm performs better than just gradient-based Gabor enhancement.

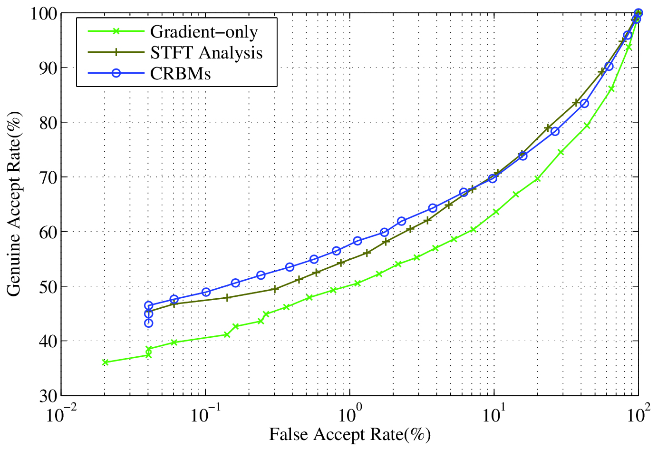

Figure 5. ROC Curves: Genuine Accept Rate plotted vs False Accept Rate. It is seen that the proposed approach is an improvement over gradient-only. As we are only interested in the relative performance, we have used the default parameters of the bozorth3 matcher. The ROC curves shown here can be improved using a better matcher.

We also calculated the number of spurious and missing minutiae generated by our algorithm on the FVC2002 Db3_a database, comparing it with hand-marked minutiae data. We observed that CRBMs reduce the number of spurious and missing minutiae detected by the gradient-only method by 9%. The proposed approach also works better than STFT-analysis [4] in reducing the number of spurious minutiae. The total number of detected minutiae is also reduced. We report in table I, the spurious and missing minutiae count:

| Method | Detected | Spurious | Missing |
|---|---|---|---|
| **Gradient-only** | 53389 | 38415(71.95%) | 4058(21.32%) |
| **STFT-analysis** | 48963 | 33096(67.59%) | 3165(16.63%) |
| **CRBMs** | 41764 | 26259(62.87%) | 3592(18.87%) |

Table I
A COMPARISON OF THE NUMBER OF SPURIOUS AND MISSING MINUTIAE DETECTED BY THREE ALGORITHMS: GRADIENT-ONLY [3], STFT-ANALYSIS [4], AND THE PROPOSED METHOD. THE TOTAL NUMBER OF MINUTIAE AS INDICATED BY GROUND TRUTH WAS 19032.

## V. CONCLUSION

We have demonstrated the use of continuous restricted Boltzmann machines in correcting orientation fields that have been estimated by a gradient method. We have moved a bit further in the direction of developing deep learning methods for orientation field estimation. In the future, we would like to perform this task using deep networks, and without initial estimates of orientation fields. With the huge database of fingerprints at our disposal, this would prove to be an area worth exploring.

## REFERENCES

[1] D. Maltoni, D. Maio, A. K. Jain and S. Prabhakar, *Handbook of Fingerprint Recognition*, Springer-Verlag London Limited, ISBN: 978-1-84882-253-5

[2] H. Chen and A. F. Murray, *Continuous restricted Boltzmann machine with an implementable training algorithm*, Vision, Image and Signal Processing, IEE Proceedings, Volume 150, No. 3, June 2003, DOI: 10.1049/ip-vis:20030362

[3] L. Hong, Y. Wan and A. Jain, *Fingerprint image enhancement: algorithm and performance evaluation*, Transactions on Pattern Analysis and Machine Learning, Volume 20, Issue 8, p.777-789, 1998.

[4] S. Chikkerur, A. N. Cartwright and V. Govindaraju, *Fingerprint Enhancement using STFT Analysis*, Pattern Recognition Journal, Volume 40, Issue 1, p.198-211, 2007

[5] J. Yang, L. Liu, T. Jiang, and Y. Fan. *A modified gabor filter design method for fingerprint image enhancement*. Pattern Recognition Letters, 24(1):18051817, 2003

[6] E. Zhu, J. Yin, and G. Zhang. *Fingerprint enhancement using circular gabor filter*. In International Conference on Image analysis and recognition, Sept. 2004

[7] Volodymyr Mnih, Department of Computer Science, University of Toronto, Toronto, Ontario, November 25, 2009. `http://code.google.com/p/cudamat/`

[8] M. A. Carreira-Perpiñán and G. E. Hinton, *On Contrastive Divergence Learning*, Artificial Intelligence and Learning, 2005

[9] G. Hinton, *A Practical Guide to Training Restricted Boltzmann Machines*, Version 1

[10] G. Taylor, G. Hinton and S. T. Roweis, *Modeling human motion using binary latent variables*, Advances in Neural Information Processing Systems, MIT Press, 2006

[11] A. Mohamed and G. E. Hinton, *Phone Recognition Using Restricted Boltzmann Machines*, ICASSP 2010, DOI: 10.1109/ICASSP.2010.5495651

[12] R. Salakhutdinov, A. Mnih, and G. E. Hinton, *Restricted Boltzmann machines for collaborative filtering*, ICML 2007, DOI: 10.1145/1273496.1273596

[13] R. Salakhutdinov and G. Hinton, *Replicated softmax: An undirected topic model*, Advances in Neural Information Processing Systems, Volume 22, 2009

[14] P. Kovesi, *Peter's Functions for Computer Vision*, www.csse.uwa.edu.au/~pk/research/MatlabFns/

[15] NIST Biometric Image Software, http://www.nist.gov/itl/iad/ig/nbis.cfm