Two Adaptive Neural Networks for Image Histogram Processing[1,2]

Irwin King
king@cs.cuhk.hk

Department of Computer Science & Engineering
The Chinese University of Hong Kong
Shatin, New Territories, Hong Kong
http://www.cs.cuhk.hk

Abstract

This paper presents two *adaptive* neural networks for image histogram processing. The first is a *supervised* network for *histogram specification* using the back-propagation learning algorithm. Once the adaptation is complete, the network represents the one-step histogram transformation function of the input histogram to the desired histogram. The second is an *unsupervised* network for *histogram equalization* based on the entropy maximization principle. This network modifies the transfer function by adaptively adjusting its gain parameters according to the entropy maximization criterion. This network results in image contrast enhancement. The architecture in both of these models is a set of K-piecewise sigmoidal neural units. We derive the adaptive rules and demonstrate simulation results for both networks.

1 Introduction

Direct histogram specification is a general technique to modify the intensity profile of an image by accentuating or attenuating a specified intensity range. Practical applications for such a transformation arise when comparing two images of a scene taken by the same sensor with fixed geometric parameters where the scene illumination conditions under which each image was acquired were different, it is useful to perform image histogram specification to ensure that the model is consistent (Wong, 1977). Others have also worked on improving the accuracy of direct histogram specification for discrete values (Zhang, 1992) and using fuzzy set operations for histogram specification (Chen et al., 1995).

Given the continuous random variable, X, to denote the intensity value of an input image with the *probability density function* (*pdf*) as $p_X(x)$ satisfying (1) $0 \leq p_X(x) \leq 1$ and (2) $\int_{-\infty}^{\infty} p_X(x)dx = 1$ with a differentiable and invertible transformation function in the form of $Y = f(X)$ satisfying (1) $0 \leq p_Y(y) \leq 1$ and (2) $\int_{-\infty}^{\infty} p_Y(y)dy = 1$, we have

$$p_Y\, dy = p_X\, dx \tag{1}$$

from probability theory. For histogram equalization of X, we have

$$y = f(x) = \int_0^x p_X(s)ds \tag{2}$$

where $f(\cdot)$ is the *cumulative distribution function* (*cdf*) of X. It can be shown from Eq. (2) that $\frac{dy}{dx} = p_X$ so when this is substituted back to Eq. (1), we obtain $p_Y = p_X \frac{dx}{dy} = p_X \frac{1}{p_X} = 1$ which is uniformly distributed between the interval $[0,1]$. Suppose $p_Z(z)$ is the desired *pdf* and the associated histogram equalization function for Z denoted by $g(Z)$, the output histogram specification is achieved by $g^{-1}(f(X))$ where $g^{-1}(\cdot)$ is the inverse transform of the histogram equalization function $g(Z)$ (Gonzalez and Woods, 1992). This is typically performed in several steps. First, find the histogram equalization of $f(X)$. Second, find the inverse histogram equalization transform of the desired output $g^{-1}(Z)$. Last, apply both functions in succession to obtain $g^{-1}(f(X))$, the desired *pdf*, p_Z. One special case is when the desired histogram is already equalized. The histogram specification is then equivalent to histogram equalization. In image processing, histogram equalization is a general technique for contrast enhancement. However, calculating *cdf* for each image is computationally intensive. An adaptive network will be able to generalize to enhance contrast of different images quickly.

We will show two ways to process image histograms using neural networks. One way is to use *supervised* learning to specify a histogram by presenting a desired histogram to the network. Another way is to use the

[1]This work was supported in parts by the Hong Kong Research Grant Council, NO: 220500720 and 220500910.
[2]A portion in this manuscript has been presented in (King and Xu, 1995).

unsupervised learning approach to achieve the desire result based on entropy maximization (King and Xu, 1995).

2 Adaptive Histogram Specification

2.1 K-piecewise Sigmoidal Approximation Network

Without loss of generality, we may assume the transformation function of the neural network is done via a set of parameters, $\theta_1, \cdots, \theta_m$ as $g(Y|X, \theta_1, \cdots, \theta_m)$. In general, a bounded and invertible logistic sigmoidal transfer function can be written as a linear combination of K-piecewise bounded, monotonic function as,

$$g(x, \theta) = \sum_{k=1}^{K} \frac{a_k}{1 + \exp(w_k x + b_k)}, \text{ with } a_k = \frac{\exp(\alpha_k)}{\sum_{l=1}^{K} \exp(\alpha_l)}, \theta = \{w_k, \alpha_k, b_k; k = 1, \cdots, K\} \tag{3}$$

which ensures that $a_k > 0, k = 1, 2, \cdots, K$ and $\sum_{k=1}^{K} a_k = 1$ for the function to remain bounded. To avoid duplication of interval, we also impose an ordering of b's as $b_1 < b_2 < \cdots < b_K$ and $w_k > 0, k = 1, \cdots, K$. Note that the summation of monotonically increasing (decreasing) functions yields a monotonically increasing (decreasing) function. Hence, we can ensure that Eq. (3) is a monotonically increasing function.

If $h(x) = \sum_{k=1}^{K} a_k/(1 + \exp(c_k(w_k x + b_k)))$ then c_k can be viewed as a scaling function which controls the "window" of operation that essentially defines the slope of the transfer function in each of the intervals defined by a_k as shown in Fig. 1(c). However, if w_k and b_k are coupled during adaptation then it is unnecessary to have c_k as a scaling variable. Hence, c_k is redundant and can be dropped from our formulation.

Both w_k and b_k are parameters which shift the transfer function along its input axis but each performs the shift differently as illustrated in Fig. 1(a) and (b) respectively. Parameter w_k controls the multiplicative shift (non-linear in normal scale and linear in log scale) for larger translational adjustments whereas b_k is the bias which controls the incremental shift of the operating range (linear in normal) for smaller translational adjustments. The a_k's control the interval in which each of the w_k and b_k could operate to match the targeted *pdf*. By adjusting these parameters, the network tunes the transfer function matching the specified histogram. Hence, the weights of the K-piecewise sigmoidal neural network essentially controls the shape of the transfer function for the input *pdf* to match against a desired *pdf*.

The neural network being constructed from Eq. (3) has a 1-K-1 architecture with a single input, K neurons in the hidden layer, and a single output as shown in Fig. 1(a). The weights, w_k's between the input and the hidden layer along with the bias, b_k, in the hidden layer represent the part of $h(x) = 1/(1 + \exp(w_k x + b_k))$ with a *sigmoidal* transfer function for the hidden neuron. Between the hidden units and the output, the weights are represented by a_k's with a *linear* transfer function in the output neuron. Together, the network functions like what is being described in Eq. (3).

Figure 1(e) demonstrates the linear combination of logistic hidden neurons and the linear output neuron for $K = 2$. By having the linear combination of these K-piecewise monotonic functions, one achieves a greater flexibility in adapting to the specific desired histogram profile.

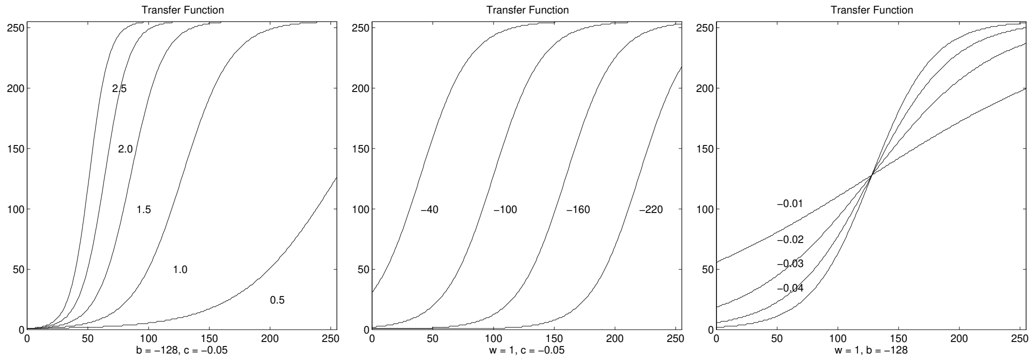
2.2 Back Propagation Learning of the Transformation Function

From the K-piecewise sigmoidal transfer function, $g(\cdot)$, we now give the adaptive rule for training the network to match the desired *pdf*. Now, let X and Y be two discrete random variables with the transformation as $Y = g(X)$. One simple way to minimize the difference of the desired, Y, with p_Y and the computed, $g(X)$, is by defining the error measure, E, based on the least squared error between X and Y as,

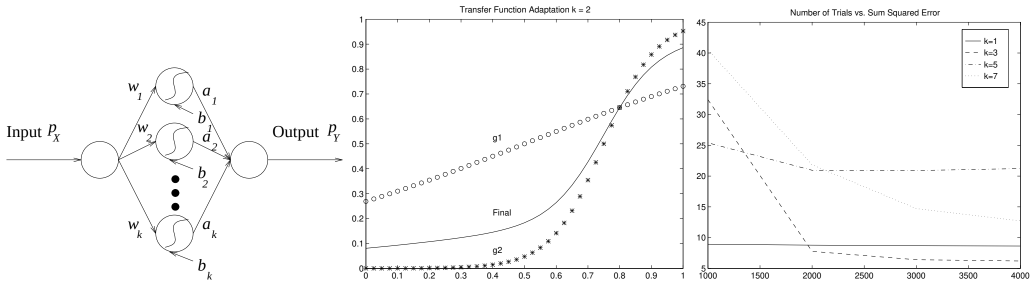
$$E = \frac{1}{2}\sum_{i=1}^{n}(y_i - g(x_i))^2 = \frac{1}{2}\sum_{i=1}^{n}\left(y_i - \sum_{k=1}^{K}\frac{a_k}{1 + \exp(w_k x_i + b_k)}\right)^2 \tag{4}$$

Eq. (4) is a convex quadratic function which contains a global minimum point at $Y = g(X)$. Hence, general gradient descent techniques are ideal in solving the above equation. The updating rules for the parameters w_k, b_k, and a_k can be obtained as,

$$\Delta w_k = -\eta_w \frac{\partial E}{\partial w_k} = \eta_w \sum_{i=1}^{n}\left(y_i - \sum_{i=1}^{K}\frac{a_i}{1 + \exp(w_k x_i + b_k)}\right)\frac{a_k x_i \exp(w_k x_i + b_k)}{(1 + \exp(w_k x_i + b_k))^2} \tag{5}$$



(a) Changing w (b) Changing b (c) Changing c



(d) 1-K-1 Network (e) Summation of 2 functions (f) Sum Squared Errors

Figure 1: (a) Adaptive sigmoidal transfer function, $b = -128$, $c = -0.05$, $w = \{0.5, 1.5, 2.0, 2.5\}$, $X = \{0, \cdots, 255\}$, (b) Adaptive sigmoidal transfer function. $w = 1$, $c = -0.05$, $b = \{-40, -100, -160, -220\}$. (c) Adaptive sigmoidal transfer function, $w = 1$, $b = -128$, $c = \{-0.1, -0.2, -0.3, -0.4\}$ (d) A 1-K-1 sigmoidal approximation neural network. (e) Summation (solid line) of two monotonically increasing functions ('o' and '*'). (f) the error obtained from trials $k = 1, 3, 5, 7$.

$$\Delta\alpha_k = -\eta_\alpha \frac{\partial E}{\partial \alpha_k} = \eta_\alpha \sum_{i=1}^{n} \left(y_i - \sum_{i=1}^{K} \frac{a_k}{1+\exp(w_k x_i + b_k)} \right) \frac{\exp(\alpha_k)\left(\sum_{l=1}^{K}\exp(\alpha_l) - \exp(\alpha_k)\right)}{(1+\exp(w_k x_i + b_k))\left(\sum_{l=1}^{K}\exp(\alpha_l)\right)^2} \quad (6)$$

$$\Delta b_k = -\eta_b \frac{\partial E}{\partial b_k} = \eta_b \sum_{i=1}^{n} \left(y_i - \sum_{i=1}^{K} \frac{a_i}{1+\exp(w_k x_i + b_k)} \right) \frac{a_k \exp(w_k x_i + b_k)}{(1+\exp(w_k x_i + b_k))^2} \quad (7)$$

2.3 Simulation Result

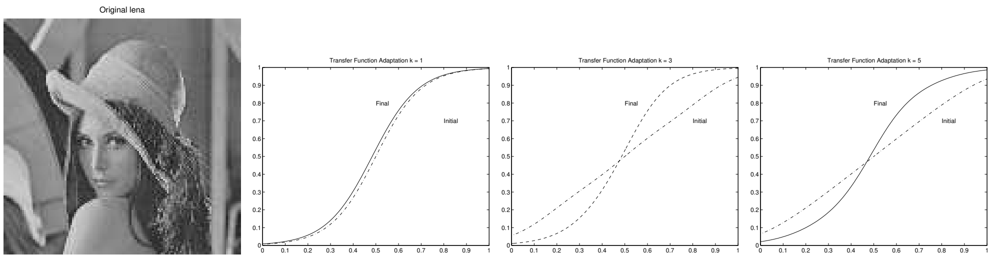
In simulating Eq. (5)-(7), we used the simple backpropagation learning algorithm with the 3-layer network shown in Fig. 1(d). The input image is a 128×128 of `lena` with $n = 256$ graylevels, i.e., $p_X(x_i)$, $i = 1, \cdots, n$. For the histogram specified image we used a histogram equalized version of `lena` for training. Hence, the final transfer function should then mimic the *cdf* of the `lena` image, i.e., see Eq. 2.

To facilitate the simulation, the training is performed via the batch mode using back-propagation where X and Y are both shown to the network at once as a single vector. For example, in our case X and Y are the $1 \times 16,384$ vectors with w, b and $a = 16,384 \times 1$ vectors.

We used a modified version of the backpropagation learning method in the neural network toolbox module in Matlab for simulation. Moreover, to simplify computation we set $a_k = 1/K$, $k = 1, \cdots, K$ in our simulation.

Table 1: The final Sum Squared Error (SSE)

K	1000 Epoch	2000 Epoch	3000 Epoch	4000 Epoch
1	8.929	8.777	8.686	8.633
3	32.42	7.762	6.41	6.202
5	25.41	20.95	20.9	21.22
7	40.51	21.83	14.72	12.72



(a) Original `lena` (b) $h(\cdot)$, $K = 1$ (c) $h(\cdot)$, $K = 3$ (d) $h(\cdot)$, $K = 5$

(e) $f(X)$ (f) $h(X), K = 1$ (g) $h(X), K = 3$ (h) $h(X), K = 5$

Figure 2: Lena images. (a) the original 128×128 lena image, (b)-(d) the initial and final transfer function found using the backpropagation learning for $K = 1$, 3, and 5 respectively, (e) histogram equalized `lena`, (f)-(h) resulting images from the transfer functions learned from BP learning for $K = 1$, 3, and 5 respectively.

A set of results for $k = 1$, 3, and 5 is shown in Fig. 2(c)-(d), (e)-(f), and (g)-(h) respectively. The final Sum Squared Error (SSE) of each k with up to 4,000 epochs is summarized in Table (1).

The 1-K-1 constrained sigmoidal network gives good approximation of the desired *pdf*. However, from the simulation it seems that the network does not get better approximation of the transfer function with more hidden neurons. For example, with $k = 5$ hidden neurons the SSE after 4000 epochs is 21.22 which is higher than when $k = 1$ or 3. Moreover, longer training might not help the network to get a better convergence as shown with the $k = 5$ case where the SSE actually increased for training of 4000 epochs than with 3000 epochs.

3 Adaptive Histogram Equalization Via Entropy Maximization

We now present our second network model based on the Maximum Entropy Principle (MEP). The MEP simply maximizes the entropy measurement subject to satisfying a set of given constraints. Here, we want to maximize the entropy function, $\mathcal{H}(Y|X, \theta)$, of the output variable Y with respect to the parameters, θ, as

$$\max_{\theta_1,\cdots,\theta_m} \mathcal{H}(Y) = \max_{\theta_1,\cdots,\theta_m} \left(-\int dy\ p_Y \log p_Y \right) = \max_{\theta_1,\cdots,\theta_m} \left(-\int dx\ p_X \log \frac{p_X}{g'(x,\theta)} \right)$$

$$
\begin{aligned}
&= \max_{\theta_1,\cdots,\theta_m} \left(-\int dx\ p_X \log p_X + \int dx\ p_X \log g'(x,\theta) \right) \\
&= \mathcal{H}(X) + \max_{\theta_1,\cdots,\theta_m} \int dx\ p_X \log g'(x,\theta) \qquad (8)
\end{aligned}
$$

Since $\mathcal{H}(X)$ is independent of θ, $\max J$ is reduced to $\max_{\theta_1,\cdots,\theta_m} \mathcal{H}(Y) \equiv \max_{\theta_1,\cdots,\theta_m} \int dx\ p_X \log g'(x,\theta) = \max_{\theta_1,\cdots,\theta_m} J$ when $J = \int dx\ p_X \log g'(x,\theta)$.

3.1 K-piecewise Sigmoidal Approximation Network

The transfer function, $g(x,\theta)$, is modeled as in Eq. (3). By having the linear combination of these K-piecewise monotonic functions, the network has a greater degree of freedom to approximate various *cdf* functions for entropy maximization of different images.

With the substitution of $g(\cdot)$, J is written as

$$
J = \int dx\ p_X \log g'(x,\theta) = \int dx\ p_X \log \left(-\sum \frac{a_k w_k \exp(w_k x + b_k)}{(1+\exp(w_k x + b_k))^2} \right). \qquad (9)
$$

We now find the derivative with respect to each of the parameter and obtain the following adaptive updating rules:

$$
\begin{aligned}
\Delta w_k &= \eta_w \Gamma \left(\frac{a_k \exp(w_k x + b_k)}{(1+\exp(w_k x + b_k))^2} - \frac{2 a_k w_k x \exp(2 w_k x + 2 b_k)}{(1+\exp(w_k x + b_k))^3} + \frac{a_k w_k x \exp(w_k x + b_k)}{(1+\exp(w_k x + b_k))^2} \right) && (10) \\
\Delta \alpha_k &= \eta_\alpha \sum_{l=1}^{K} \left(\Gamma\ \frac{w_l \exp(w_l x + b_l)}{(1+\exp(w_k x + b_k))^2}\ a_l (\delta_{k,l} - a_k) \right), a_k = \frac{\exp(\alpha_k)}{\sum_{i=1}^{K} \exp(\alpha_i)} && (11) \\
\Delta b_k &= \eta_b \Gamma \left(\frac{a_k w_k \exp(w_k x + b_k)(1 - \exp(w_k x + b_k))}{(1+\exp(w_k x + b_k))^3} \right), \text{ where} && (12) \\
\Gamma &= \frac{1}{\sum_{i=1}^{K} \frac{a_i w_i \exp(w_i x + b_i)}{(1+\exp(w_i x + b_i))^2}}. && (13)
\end{aligned}
$$

3.2 Simulation Result

The simulation used a 1-5-1 neural network. The iterative updates are calculated according to Eq. (10)-(13). Using digitized grayscale images, the continuous random variables X and Y are converted into discrete random variables with $n = 256$ states (graylevels). The image we used is 128×128 of `lena` shown in Fig. 3 (a). Its histogram, histogram equalized output, equalized histogram, and the *cdf* are illustrated in Fig. 3 (b)-(e) respectively. Results are shown in Fig. 3 (f)-(i). Fig. 3 (f) is the transformed output from the network. The quality is similar to (c) which is calculated by the standard histogram equalization technique. The transformed output's histogram in (g) which should be similar to (d) is more uniformly distributed than in (b). Furthermore, Fig. 3 (h) illustrates that the set of adaptive rules does increase the entropy at the output. Fig. 3 (i) displays the final transformation function formed by the 1-5-1 network. The larger diagonal slope from the bottom-left to the top-right is similar to the one calculated for `lena` shown in (e).

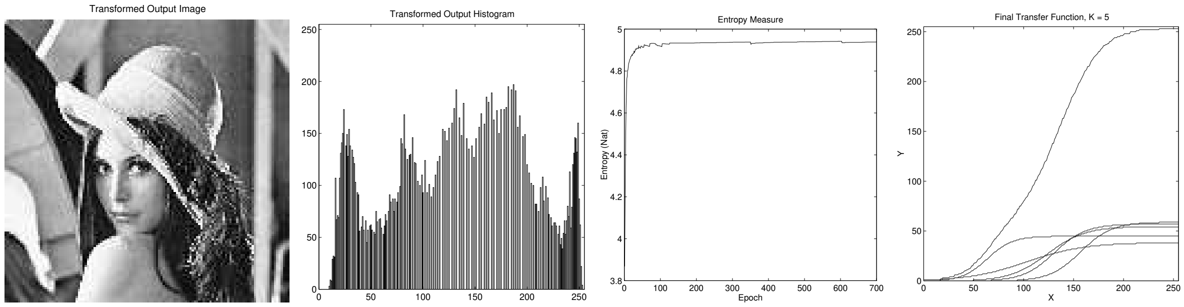
4 Conclusion

We have demonstrated here two *adaptive* 1-K-1 constrained sigmoidal neural networks, one for histogram specification using the backpropagation learning technique and the other for histogram equalization using the Maximum Entropy Principle. Experimental results demonstrate these networks perform according to the expected outcome. By processing histogram adaptively, we achieve (1) the ability to learn a specified histogram and (2) the ability to approximate and generalize *cdf* functions automatically for contrast enhancement with on-line images.

References

Chen, B.-T., Chen, Y.-S., and Hsu, W.-H. (1995). Automatic histogram specification based on fuzzy set operations for image enhancement. *IEEE Signal Processing*, 2(2):37–40.

(a) (b) (c) (d) (e)



(f) (g) (h) (i)

Figure 3: (a) original `lena`, (b) `lena`'s histogram, (c) histogram equalized `lena`, (d) histogram for equalized `lena`, (e) the Cumulative Distribution Function (*cdf*) function, (f) output image calculated from the 1-5-1 network, (g) output image's histogram, (h) entropy vs. learning steps, (i) the transfer function formed by the 1-5-1 network, in comparison with the *cdf* function of (e).

Gonzalez, R. and Woods, R. E. (1992). *Digital Image Processing.* Addison-Wesley, Reading, MA, USA.

King, I. and Xu, L. (1995). Adaptive contrast enhancement by entropy maximization with a 1-K-1 constrained network. In Zhong, Y., Yang, Y., and Wang, M., editors, *Proceedings to the International Conference on Neural Information Processing*, pages 703–706, Beijing, PRC. Publishing House of Electronics Industry.

Wong, R. Y. (1977). Sensor transformations. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-7(12):836–841.

Zhang, Y. J. (1992). Improving the accuracy of direct histogram specification. *Electronics Letters*, 28(3):213–214.

Author

Irwin King, Assistant Professor of Department of Computer Science & Engineering, The Chinese University of Hong Kong. His specialty is in the area of neural networks, image processing, and unsupervised visual learning.