J.M.Bishop & M.J.Bushnell

Neural Network Research Group,

Department of Cybernetics,

Reading University,

Whiteknights,

Reading, Berkshire, RG6 2AX,

England.

S.Westland

Vision Group,

Department of Communication & NEuroscience,

University of Keele,

Staffordshire ST5 5BG,

England.

The Application of Neural Networks to Computer Recipe Prediction

Abstract

Conventional mechanisms for computer colorant formulation commonly employ the Kubelka-Munk theory to relate reflectance values to colorant concentrations, however there are situations where this approach is not applicable and hence an alternative is desirable. One such method is to utilise Artificial Intelligence techniques to mimic the behaviour of the professional colourist. The purpose of this paper is to describe recent research being carried out at Reading University, sponsored by Courtaulds Research, that utilises collections of cellular automata, known as Neural Networks, in the problem of recipe prediction.

Introduction

Since the early development of a computer colorant formulation method computer recipe prediction has become one of the most important industrial applications of colorimetry. The first commercial computer for recipe prediction[1] was an analog device known as the COMIC (*co*lorant *mi*xture *c*omputer) and this was superseded by the first digital computer system, a Ferranti Pegasus computer, in 1961[2].

Several companies today market computer recipe prediction systems and all are based on digital computers. All computer recipe prediction systems developed to date are based on an optical model that performs two specific functions:

- The model relates the concentrations of the individual colorants to some measurable property of the colorants in use;

- The model describes how the colorants behave in mixture.

The model that is commonly employed is the Kubelka-Munk theory which relates measured reflectance values to colorant concentration via two terms *K* and *S*, which are Kubelka-Munk versions of the absorption and scattering coefficients respectively of the colorant. In order for the Kubelka-Munk equations to be used as a model for recipe prediction it is necessary to establish the optical behaviour of the individual colorants as they are applied at a range of concentrations. It is then assumed that the Kubelka-Munk coefficients are additive when mixtures of the colorants are used. Thus it is usual for a database to be prepared which includes all of the colorants which are to be used by the system and allows the calculation of *K* and *S* for the individual colorants.

The Kubelka-Munk theory is in fact an approximation of an exact theory of radiative transfer. Exact theories are well documented in the literature[3], but have rarely been used in the coloration industry. The Kubelka-Munk approximation is a two-flux version of the many-flux treatment for solving radiative problems.

In order for the Kubelka-Munk approximation to be valid the following restrictions are assumed:

- The scattering medium is bounded by parallel planes and extends over a region very large compared to its thickness;

- The boundary conditions which include the illumination, do not depend upon time or the position of the boundary planes;

- The medium is homogenous for the purposes of calculation;

- The radiation is confined to a narrow wavelength band so that the absorption and scattering coefficients are constant;

- The medium does not emit radiation (e.g. fluoresce);

- The medium is isotropic.

There are many applications of the Kubelka-Munk approximation in the coloration industry, where these assumptions are known to be false. In particular, the applications to thin layers of colorants (e.g. printing inks[4]) and fluorescent dyestuffs[5,6] have generally yielded poor results.

The use of an approximation of the exact model has attracted criticism. For example, Van de Hulst[7] when discussing its application to paint layers comments: "*it is a pity that all this work has been based on such a crude approximation...*". The popularity of the Kubelka-Munk equations are undoubtedly due to their simplicity and ease of use. The equations give insight and can be used to predict recipes with reasonable accuracy in many cases. In addition, the simple principles involved in the theory are easily understood by the non-specialist and rightly form the basis for study for those involved in the coloration industry[8].

The use of the exact theory of radiation transfer is not of practical interest to the coloration industry. The calculations generally require data bases and spectrophotometers of a greater complexity than those suitable for Kubelka-Munk calculations.

The performance of the Kubelka-Munk theory in certain areas of coloration is such to warrant an alternative approach. It is possible for professional colourists to perform recipe prediction to a high standard even for fairly complex situations and yet such colourists do so without knowledge of the Kubelka-Munk theory or indeed any optical model. The traditional colourist accumulates experience of the behaviour of the colorants and is able to extrapolate and interpolate from this data to predict recipes for new shades. Computer hardware and software technology is such that it is now possible to mimic the performance of human operators or '*experts*' in many areas of science and technology. One particular computer technology that could be used for recipe prediction is that of neural networks.

What is a Neural Network ?

A Neural Network can be considered a black box, which is connected to the world by a series of inputs and which interacts with the world via a set of outputs. The task of the Network is to perform a set of mappings between its input and output. What makes Neural Networks unusual and powerful is their ability to deal with fuzzy real world data, and their potential for performance improvement over time as they acquire more knowledge about a problem.

Unlike conventional computer solutions to specific problems a neural network is not explicitly programmed to complete a given task, rather it adapts and acquires knowledge over time in order to complete it. It is this power that enables the use of network models on computationally ill-defined systems.

Parallel Distributed Processing (PDP) techniques

Neural Networks consist of collections of connected processing elements that are not individually programmable. Each cell usually computes a simple non linear function f on the weighted sum of its input (figure 1). The output of this function is defined as the activation of the cell. Long term knowledge is stored in the network in form of interconnection weights linking these cells.

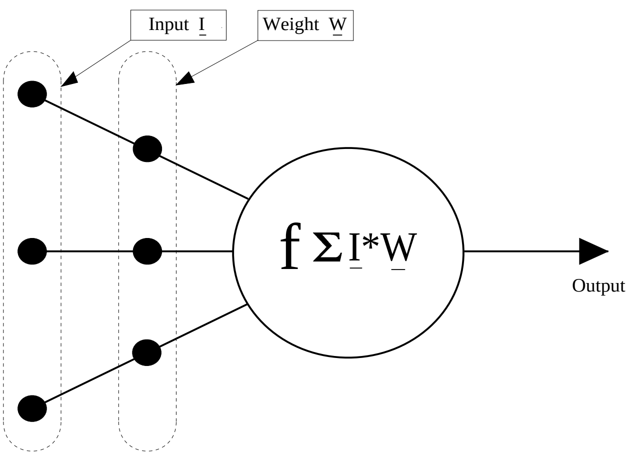

FIG 1. The generalized neuron model

The Neural Network used in these experiments consisted of an input layer where cell inputs are clamped to external values (eg. scaled CIELAB values). Two hidden layers, where cell inputs are defined by the weighted activation values from the cells in the previous layer that they are connected to, and an output layer connected to the second hidden layer (see figure 2). On presentation of input, the network will rapidly settle into a stable state consistent with it.

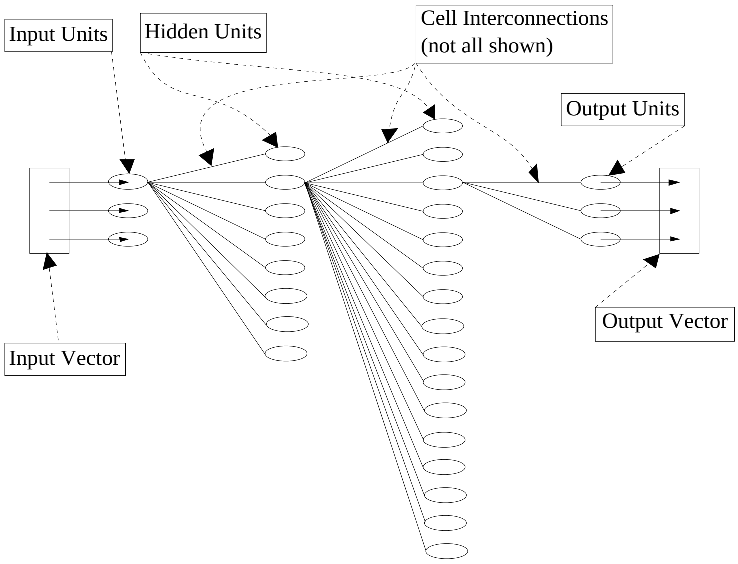

FIG 2. Multilayer neural network

In a Neural Network, learning is the process of defining a set of weights that will produce a desired response to an input pattern. Two of the best known learning strategies for multi-layer connectionist machines are The Boltzmann machine learning algorithm[9] and Back Propagation of the Generalised Delta Rule[10], the latter of which has been used in this study.

Given enough cells and having learnt the right set of inter-cell weights, a network of this sort can produce input / output mappings of arbitrary complexity and can find them quickly as all of its knowledge operates in parallel. The process of finding the optimum set of inter-cell weights is very computationally intensive, however this initial learning phase is only executed once. These experiments used an IBM compatible 12MHz ' 286 machine, where typical timings for learning were around 48 hours, whereas in use the network took less than a tenth of a second to predict a recipe.

Neural Networks and Recipe Prediction

One of the main problems with recipe prediction is that the application of exact colour theory is not computationally practical and an approximation to it has to be employed. It was expected that a Neural Network approach to recipe prediction would offer a novel and profitable new solution to this problem, since many problems in Artificial Intelligence (AI) involve systems where conventional rule based knowledge is not perfect or the application of the pure theory is too computer intensive to be used in practical systems. Neural Networks are already widely used in Computer Vision, both in the investigation of difficult problems such as facial feature location[11,12] and in practical general purpose industrial vision systems[13].

In the field of recipe prediction it was hoped that a suitable network system would automatically learn relationships between colorants and colour, and hence learn to predict which colorants, and at which concentrations, need to be applied to a particular substrate in order to produce a specified colour.

The Back Propagation, or Generalised Delta, Learning Rule

Many different learning rules have been developed for neural networks. One of the key developments in neural network research in the past decade had been the development of learning rules that can teach hidden units. One such rule is the Generalised Delta Rule, which propagates an error term backwards through the net from the output units to the input units making weight changes such that overall error in the net is minimised. A complete introduction to this learning rule[10] is beyond the scope of this paper, however a brief description will be given.

The generalised delta rule works by performing gradient descent in Error/Weight space. That is, after each pattern has been presented, the resulting error on that pattern is computed, by comparing the actual output with the desired output, and each weight in the network modified by moving down the error gradient towards its minimum for that input/output pattern pair. Gradient descent involves changing each weight in proportion to the negative of the derivative of the error, defined by this pair.

The Generalised Delta Rule is based on the original Delta or Widrow-Hoff learning rule[14] . The weight update procedure for this rule, given an input vector I, an output vector O, a target vector T and a weight matrix W is:

$$\Delta W_{ji} = \eta\,(t_i - o_j)\,i_i \qquad (1)$$
$$= \eta\,\delta_j\,i_i$$

Where ΔW_{ji} is the change to be made to the weight linking the i_{th} to the j_{th} unit, given the current input/output vector pair. η is defined as the learning rate constant. δ_j can be considered as an error term describing the difference between the desired output and the actual output. The key development, by the PDP research group at the University of California, was the extension of the above learning rule for single layer networks, to function with multi-layer nets.

The basic form of the Generalised Delta Rule is identical the the Simple Delta Rule:

$$\Delta W_{ji} = \eta\,\delta_j\,i_i \qquad (2)$$

At the output layer the error signal is:

$$\delta_j = (t_j - o_j)\,f_j'(net_j) \qquad (3)$$

and for a hidden layer the error signal is:

$$\delta_j = f_j'(net_j) \times \sum_{k=1} \delta_k W_{kj} \qquad (4)$$

Where $f_j'(net_j)$ is the derivative of a semilinear activation function acting on the jth unit, which maps the total input to the unit to an output value. A semilinear function must be differentiable, continuous, monotonic and non-linear. The activation function used in this study is the logistic activation function:

$$o_j = \frac{1}{(1 + e^{-net_j})} \qquad (5)$$

Where $net_j = (\sum_i W_{ji} \times o_i + \theta_j)$. θ_j is a bias term and can be learned like any other weight by considering it connected to a unit which is permanently on. The derivative of f is thus:

$$\frac{\partial o_j}{\partial net_j} = o_j(1 - o_j) \qquad (6)$$

Thus for an output unit the error signal is:

$$\delta_j = (t_j - o_j)\,o_j(1 - o_j) \qquad (7)$$

and for a hidden unit the signal is:

$$\delta_j = o_j(1 - o_j) \times \sum_{k=1} \delta_k W_{kj} \qquad (8)$$

Learning Rate η

The above learning procedure requires only that the change in weight is proportional to $\frac{\partial E}{\partial W}$, whereas true gradient descent requires infinitesimal small steps to be taken. The constant of proportionality is the learning rate constant, η. The larger this value, the larger the changes in weight at each iteration and the faster the network learns. However if the learning rate is too large, then the network will go unstable and oscillate. It has been shown[15] that one way to increase the learning rate, without leading to oscillation is to introduce a momentum term. ie.

$$\Delta W_{ji}(n + 1) = \eta(\delta_j o_i) + \alpha \Delta W_{ji}(n) \qquad (9)$$

where n indexes the current input/output presentation number, η is the learning rate constant and α is the momentum constant. This defines how much past weight changes effect the current direction of movement, providing a force analogous to momentum in weight space, which acts to filter out high frequency variations in the error surface. For example, if the network arrives at a gradually descending ravine in weight error space, the steepest error gradient may be mainly across, rather than down, the ravine. The use of a momentum term tends to filter out such sideways movement, while compounding movement down the ravine.

In one learning sweep with the Generalised Delta Rule, the following sequence of events occur:

- The network is presented with an input pattern (all the input units of the network are set to the required values).
- This input vector is used to compute the output values by feeding forward through the net and computing activation values for all the other units according to the weight values.
- The output vector for this pattern is compared to the required, or target, pattern and the error term is calculated for every output unit.
- Error terms are recursively propagated backwards through the net to the other units in proportion to the connection strengths between the units.
- The weights are then adjusted in such a way as to reduce the error terms, by performing gradient descent in weight error space, in a similar manner to the Simple Delta Rule.
- The process is repeated for all the input/target pattern pairs in the training set.

The process of presenting to the network all the patterns over which it is to be trained is defined as an epoch. Training continues for as many epochs as are necessary to reduce the overall error to an acceptably low value.

Experimental Results

In the experiments described below the input data (CIELAB coordinates) were used in LCH and LAB form and after scaling to the interval [0..1], directly applied to three input units.

The dye system used for these initial tests consisted of three dyes on a nylon substrate. Because the aim of the experiment was to investigate the feasibility of recipe prediction using a neural network it was not necessary to use real data and hence the training and test recipes were synthesised using an ICS-TEXICON colour system. The ICS-Texicon system included a conventional recipe prediction data base and was capable of predicting recipes and colour coordinates using a standard Kubelka-Munk model. The three dyes chosen were Nylosan Blue (EGL), Nylosan Yellow (ERPL) and Nylosan Scarlet (F-3GL) and recipes were synthesised using varying concentrations of single and two dye mixtures from this group, to produce CIELAB colour coordinates under D65 illuminant data.

The neural network consisted of three input units, 24 hidden units arranged in two layers of 8 and 16 units, and three output units (figure 2). The input units correspond to the three colour co-ordinates and the output units to concentrations of the three dyes. The net was trained using a low learning rate as empirical evidence showed that the network was unstable at higher rates. The net was trained on approximately half of the synthesised recipes with the remaining recipes being used as test data in order to discover whether the net could produce accurate recipes for previously unseen colours. The results were evaluated using the ICS system mentioned above, using the CMC (2:1) colour difference equation.

a: Experiments using data in LCH format

The first tests performed used data in LCH format trained over 3000 epochs. Observation of the network during training showed that after some initial progress the network ceased learning and training was aborted. Examination of the results illustrates particularly poor performance for the blue and scarlet mixture recipes (see figure 3a). This is caused by the nature of the hue angle: values close to 360 degrees are similar in hue to angles close to 0 degrees. Thus colours that are almost identical can have a disproportionately large variation in the hue angle input to the net.

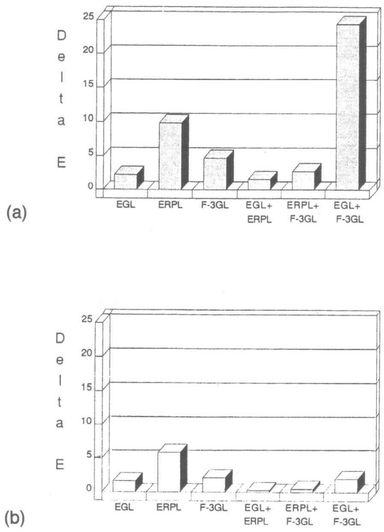

FIG. 3. Average ΔE. (a) LCH format. (b) LAB format.

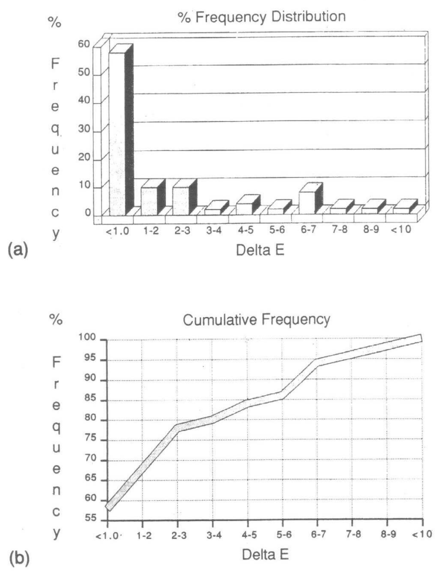

FIG. 4. Combined performance.

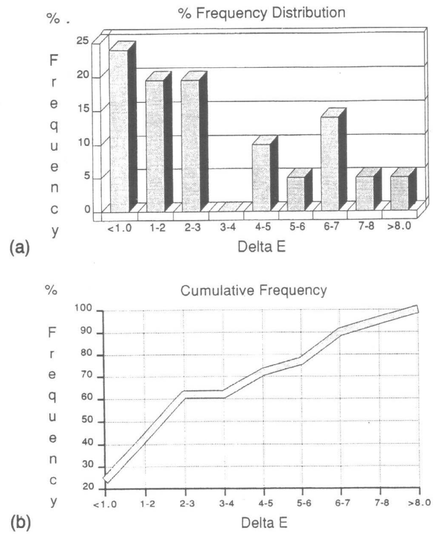

FIG. 5. Single-dye performance.

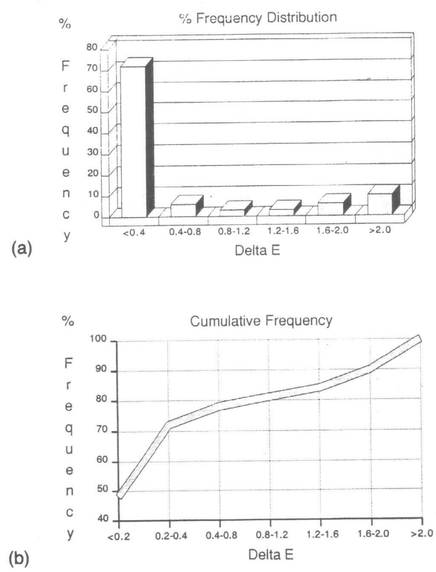

FIG. 6. Two-dye mixture performance.

b: Experiments using data in CIELAB format

LAB tests used the same data set and conditions as the LCH tests, but with the input in LAB format instead of LCH. Observation of the network during training showed a gradual decrease in error over time. Training was continued for 55000 epochs until improvement became marginal. The results are given in Table One and are summarised in figure 3b.

			Actual % dye conc.			Values produced by net			
L	*A*	*B*	EGL	ERPL	F-3GL	EGL	ERPL	F-3GL	Δ*E*
				Blue only					
*52.52	−9.03	−34.73	0.500	0.000	0.000	0.510	0.040	0.000	8.10
45.07	−6.85	−37.16	1.000	0.000	0.000	1.080	0.010	0.000	1.60
*37.52	−4.28	−37.84	2.000	0.000	0.000	1.990	0.000	0.000	0.04
34.84	−3.40	−37.49	2.500	0.000	0.000	2.470	0.000	0.000	0.09
*32.66	−2.72	−37.03	3.000	0.000	0.000	3.020	0.000	0.000	0.05
30.83	−2.17	−36.53	3.500	0.000	0.000	3.550	0.000	0.010	0.80
*29.26	−1.73	−36.03	4.000	0.000	0.000	3.990	0.000	0.010	0.64
				Yellow only					
*84.46	2.10	64.16	0.000	0.500	0.000	0.000	0.500	0.010	7.3
81.75	6.55	73.20	0.000	1.000	0.000	0.000	0.990	0.010	5.7
*79.63	10.78	79.72	0.000	2.000	0.000	0.000	1.960	0.010	4.6
78.77	12.47	81.88	0.000	2.500	0.000	0.000	2.530	0.010	4.3
*77.99	13.95	83.60	0.000	3.000	0.000	0.000	3.080	0.020	6.8
77.29	15.27	85.02	0.000	3.500	0.000	0.000	3.560	0.020	6.4
*76.65	16.45	86.19	0.000	4.000	0.000	0.000	3.940	0.020	6.1
				Scarlet only					
*58.25	56.03	41.39	0.000	0.000	0.500	0.000	0.080	0.520	1.9
54.25	58.94	47.16	0.000	0.000	1.000	0.010	0.070	1.100	2.9
*49.70	59.03	48.92	0.000	0.000	2.000	0.010	0.050	2.000	2.4
47.76	58.64	49.00	0.000	0.000	2.500	0.010	0.050	2.520	2.2
*46.13	58.26	48.99	0.000	0.000	3.000	0.010	0.050	3.040	2.0
44.73	57.88	48.92	0.000	0.000	3.500	0.010	0.060	3.530	1.9
*43.50	57.51	48.80	0.000	0.000	4.000	0.010	0.080	3.950	1.7
				Blue and Yellow					
*48.10	−26.00	6.55	0.500	0.500	0.000	0.480	0.500	0.000	0.51
*41.31	−24.77	−2.52	1.000	0.500	0.000	1.010	0.490	0.000	0.30
40.01	−25.96	6.75	1.000	1.000	0.000	1.010	1.030	0.000	0.12
46.64	−25.80	16.31	0.500	1.000	0.000	0.470	1.070	0.000	1.10
33.35	−23.83	−2.06	2.000	1.000	0.000	1.990	1.040	0.000	0.30
*32.37	−24.55	4.49	2.000	2.000	0.000	1.990	2.020	0.000	0.13
38.88	−25.51	13.71	1.000	2.000	0.000	1.010	1.990	0.000	0.12
*29.15	−21.20	−6.95	3.000	1.000	0.000	3.000	1.000	0.000	0.00
28.27	−22.64	−0.79	3.000	2.000	0.000	3.070	2.140	0.000	0.36
*31.63	−24.34	8.84	2.000	3.000	0.000	2.000	2.990	0.000	0.03
*38.04	−24.49	18.23	1.000	3.000	0.000	1.000	3.000	0.000	0.00
				Yellow and Scarlet					
*57.63	54.20	51.56	0.000	0.500	0.500	0.000	0.470	0.480	0.24
*57.31	53.52	56.20	0.000	1.000	0.500	0.000	1.000	0.490	0.11
53.55	57.22	56.45	0.000	1.000	1.000	0.000	1.060	1.030	0.11
53.80	57.75	53.22	0.000	0.500	1.000	0.000	0.460	1.040	0.26
*53.38	56.88	59.54	0.000	2.000	1.000	0.000	2.010	1.040	0.11
49.09	57.66	57.03	0.000	2.000	2.000	0.000	2.020	1.950	0.14
*49.21	57.92	54.78	0.000	1.000	2.000	0.000	0.980	1.930	0.14
53.22	56.61	61.89	0.000	3.000	1.000	0.000	2.920	1.050	0.25
*48.97	57.45	58.82	0.000	3.000	2.000	0.000	3.010	2.000	0.01
45.68	57.27	54.84	0.000	2.000	3.000	0.010	2.110	3.110	2.00
*45.77	57.46	53.12	0.000	1.000	3.000	0.010	0.980	3.050	2.00
				Blue and Scarlet					
*32.90	15.71	−0.62	0.500	0.000	0.500	0.500	0.000	0.490	0.26
*28.48	10.66	−7.14	1.000	0.000	0.500	1.000	0.000	0.510	0.21
25.72	13.59	−0.47	1.000	0.000	1.000	1.090	0.000	1.060	0.67
30.01	19.16	6.29	0.500	0.000	1.000	0.580	0.000	1.010	1.40
*21.60	9.01	−6.52	2.000	0.000	1.000	2.000	0.000	1.000	0.00
19.69	10.33	−1.29	2.000	0.000	2.000	2.000	0.000	1.400	3.50
*23.61	15.45	4.84	1.000	0.000	2.000	1.010	0.000	2.000	0.10
19.10	6.71	−9.78	3.000	0.000	1.000	3.210	0.000	0.550	6.00
*17.31	7.64	−4.74	3.000	0.000	2.000	3.000	0.010	2.000	0.11
18.31	11.60	2.05	2.000	0.000	3.000	2.346	0.000	1.450	9.30
*22.09	17.00	8.14	1.000	0.000	3.000	0.990	0.000	3.000	0.18

*Patterns presented during training.

TABLE 1. Results of network prediction using LAB input data.

From analysis of the results it was clear that the LCH format is not suitable for use with this type of network and all further experimentation has used the LAB format for colour measurement.

Conclusion

Initial results, using a simple Multi-Layer Perceptron Neural Network model, have demonstrated the feasibility of using neural networks for computer recipe prediction and indicate that the technique is worth investigating further. Although LCH colour definition was unsatisfactory for this research (figure 3), approximately 60% of all predictions using LAB format data result in ΔE values less than one (see figure 6).

The results for single dye predictions are relatively poor, only 24% of predictions result in a ΔE value less than one (figure 4). This may be due to problems at the extremities of the concentration scaling, since by definition in one dye formulations the concentrations of the other two dyes are at an extreme (0.00). In particular, the results for the blue single dye mixtures are significantly better between the two concentration extremes [0.5 .. 4.0]. Computationally, a likely cause for this phenomena is the use of the sigmoid logistic activation function, which results in values at the extremities of the output range becoming hard to learn[14]. Work is ongoing to resolve this problem.

The two dye predictions are very promising. 78.8% of all such predictions result in ΔE values less than 0.8 (figure 5). Research is continuing to see if this performance can be achieved on larger scale experiments.

The work reported in this paper suggests that neural network techniques maybe be useful for solving recipe prediction problems. It has been shown that the Kubelka-Munk model has been approximated without any a priori knowledge about the system. There is no reason to believe that similar neural networks cannot learn the relationship between colorant concentrations and colour coordinates for real coloration systems. In order for this approach to become viable it will be necessary to extend the training data to include a greater number of colorants. It remains to be seen how many training recipes will be necessary to enable the network to make accurate predictions when a larger number of colorants are used. It will also be necessary to be able to include information regarding illuminants other that D65; it is possible that this may be accomplished by entering colorimetric data for more than one illuminant or by using reflectance values for the target data during training.

The use of neural networks offers several potential advantages over the conventional Kubelka-Munk approach:

- It is not necessary to prepare a special database in order to use the neural network method. Conventional Kubelka-Munk systems necessitate data base samples to be produced and up to ten samples per colorant is not unusual. The network would be trained on actual production samples. Most dyehouses, for example, maintain historical shade data and this would be most suitable for training.

- The neural network is able to continue to learn after the initial training period, since future production samples can be presented to the system and this knowledge incorporated into the network weights. This gives the network the potential to adapt to changes in important factors such as water supply, substrate properties or colorant colour strengths in the same way that a colourist would adapt to such changes over time.

- The neural network approach may be able to learn the behaviour of colorants for coloration systems for which the mathematical descriptions are complex. For example, fluorescent dyes and metallic paint systems, are currently difficult to treat using standard Kubelka-Munk theory.

The application of neural networks to recipe prediction represents a radical departure from conventional mathematical treatments and this initial study suggests that the approach may be successful.

References

[1] Davidson, H.R., Hemmendinger, H. & Landry, J.L.R., *A System of Instrumental Colour Control for the Textile Industry*, Journal of the Society of Dyers and Colourists: **79**, pp. 577, (1963).

[2] Alderson, J.V., Atherton, E. & Derbyshire, A.N. *Modern Physical Techniques in Colour Formulation*, Journal of the Society of Dyers and Colourists: **77**, pp. 657-668, (1961).

[3] Chandrasekhar, S., *Radiative Transfer*, Clarendon Press, Oxford, (1950).

[4] Westland, S., *The Optical Properties of Printing Inks*, PhD Thesis, Leeds University (UK), (1988).

[5] Ganz, E., *Problems of Fluorescence in Colorant Formulation*, Colour Research and Application: **2**, pp. 81-84, (1977).

[6] McKay, D.B. *Practical Recipe Prediction Procedures including the use of fluorescent dyes*, PhD Thesis, University of Bradford (UK), (1976).

[7] Van de Hulst, H.C., *Multiple Light Scattering; Tables, Formulas and Applications*, Academic Press, New York, (1980).

[8] Nobbs, J.H., *Review of Progress in Coloration*, The Society of Dyers and Colourists, Bradford, (1986).

[9] Hinton, G.E., Sejnowski, T.J., & Ackley, D.H., *Boltzmann networks: Constraint Satisfaction networks that learn*, Technical Report No. CMU-CS-84-119, Carnegie-Mellon University, Dept. of Computer Science, (1984).

[10] Rumelhart, D.E., Hinton, G.E., & Williams, R.J., *Learning Internal Representations by Error Propagation*, in Rhumelhart & McLelland, *PDP: Explorations in the Micro-Structure of Cognition: **1***, MIT Press, (1986).

[11] Bishop, J.M., *Anarchic Techniques for Pattern Classification*, PhD Thesis, Reading University (UK), (1989).

[12] Bishop, J.M., *Stochastic Searching Networks*, Proc. 1st IEE conf. on Artificial Neural Networks. IEE Conf. Publication No. 313, pp. 329-331, (1989).

[13] Aleksander, I., Thomas, W.V. & Bowden, P.A. *WISARD: A Radical Step Forward in Image Recognition*, Sensor Review: **4(3)**, pp. 120-124, (1984).

[14] Widrow, G. & Hoff, M.E., *Adaptive switching circuits*, I.R.E. Western Electronics Snow & Convention, Convention Record (Part 4), pp. 96-104, 1960.

[15] Rumelhart, D.E., McLelland, J.L. & PDP Research Group, *PDP: Explorations in the Micro-Structure of Cognition:* ***1***, pp. 329, MIT Press, (1986).