01 May 1993

Diagnosis of Malignant Melanoma using a Neural Network

Anurag Chawla

Fikret Erçal
Missouri University of Science and Technology, ercal@mst.edu

DIAGNOSIS OF MALIGNANT MELANOMA USING A NEURAL NETWORK

A. Chawla* and F. Ercal

CSc-93-06

Department of Computer Science

University of Missouri - Rolla

Rolla, MO 65401 (314)341-4491

*This report is substantially the M.S. thesis of the first author, completed May 1993.

ABSTRACT

Malignant melanoma is the deadliest form of all skin cancers. Approximately 32,000 new cases of malignant melanoma were diagnosed in 1991, with approximately 80 percent of patients expected to survive five years [1]. Fortunately, if detected early, even malignant melanoma may be treated successfully. Thus, in recent years, there has been a rising interest in the automated detection and diagnosis of skin cancer, particularly malignant melanoma [2]. In this thesis, a novel neural network approach for the automated distinction of melanoma from three benign categories of tumors which exhibit melanoma-like characteristics is presented. The approach is based on devising new and discriminant features which are used as inputs to an artificial neural network for classification of tumor images as malignant or benign. Promising results have been obtained using this method on real skin cancer images.

TABLE OF CONTENTS

Page

Page

LIST OF TABLES

I. INTRODUCTION

A. Skin Cancer Characteristics

Dermatology imaging researchers believe that diagnosis of skin tumors can be automated based on certain physical features and color information that are characteristic of the different categories of skin cancer. Diagnosis of malignant melanoma is a difficult task since other skin cancers have similar physical characteristics. In many cases, dermatologists must perform a biopsy (a laboratory medical procedure) to ascertain whether a tumor is malignant or benign. Since this is a costly procedure, alternative early detection techniques are being sought to use as an adjunct for rapid inexpensive skin cancer screening. In this study, we use color images of skin tumors and an artificial neural network to distinguish melanoma from other benign pigmented tumors: dysplastic nevi, intradermal nevi and seborrheic keratoses. We first define those features that are expected to distinguish melanoma from three other skin tumors, and train an artificial neural network with these features in an attempt to classify the tumor type as melanoma or not. The characteristics of malignant melanoma and three other categories of benign tumors which are difficult to distinguish from melanoma are outlined below. These descriptions apply to only the most typical members of a diagnostic group.

1. Malignant Melanoma (mel). Malignant melanoma is named for the cell from which it presumably arises, the melanocyte. Melanocytes are the skin cells which produce the dark protective pigment called melanin, a natural sunscreen. Melanoma cells

usually continue to produce melanin, which accounts for the cancers appearing in mixed shades of tan, brown and black (variegated coloring). Melanoma has a tendency to metastasize (spread), hence early detection and treatment are essential. Friedman et al. have enumerated the mnemonic "ABCD" to describe early malignant melanoma [3]:

• **Asymmetry** - One half of the tumor does not match the other half.

• **Border Irregularity** - The edges are ragged, notched, blurred.

• **Color** - Pigmentation is not uniform. Shades of tan, brown and black are present. Dashes of red, white and blue add to the mottled appearance.

• **Diameter** - greater than 6 mm and growing.

2. Dysplastic Nevi (dys nevi). Moles, or nevi, are tan brown spots on the skin that result from a clustering of melanocytes. Certain unusual moles called dysplastic nevi are likely to undergo changes leading to melanoma. Scientists believe that individuals with dysplastic nevi, especially those from families with multiple cases of melanoma represent one group of people who are more likely to develop melanoma. It is important to remember that, although the dysplastic nevus is the kind of mole most likely to undergo malignant changes, most dysplastic nevi do not become malignant. The National Cancer Institute [4] has outlined the following characteristics for the detection of dysplastic nevi, lesions that may occur in both familial and non-familial settings, and are associated with a higher risk of malignant melanoma:

• **Color** - Mixture of tan, brown, black and red/pink.

• **Shape** - Irregular Borders that may include notches. May fade into surrounding skin and include a flat portion level with the skin.

• **Surface** - Smooth, slightly scaly, or have a rough pebbly appearance.

• **Size** - often larger than 5mm and sometimes larger than 10mm.

3. Intradermal Nevi (Idn). This is a benign tumor. Idn is most common in children and young adults and may be tan, brown, flesh or pink. These are commonly called moles and may be hairy. Dermatologists agree upon the following characteristics of idn (modified from [4]):

• **Color** - Flesh colored, pink, may be tan or brown.

• **Shape** - Round or oval, may fade gradually into the surrounding skin.

• **Surface** - Often smooth, sometimes papillomatous, and raised. Skin markings are present when examined with a hand lens.

• **Size** - Usually less than 6mm in diameter.

4. Seborrheic Keratoses (sk). This is a benign tumor found in older persons, with patients usually older than forty. It is a benign growth of the epidermis (outer layer of the skin) with the following clinical characteristics (modified from [1]):

• **Color** - Tan to brown, may be fleshy or pink, darker in persons with darker skin.

• **Shape** - Borders often oval or round but may be irregular, often sharply demarcated but in fair persons fading gradually into surrounding skin.

• **Surface** - Rough, verrucous, sometimes with keratin plugs. Skin markings are almost always enhanced, even if the surface is not rough. The raised surface and frequently sharp border lead to the appellation "stuck-on". The tan to yellowish color combined with the stuck-on appearance is sometimes called "tallow-drop."

• **Size** - 3mm - 30mm or more, usually 5-15 mm.

• **Location** - Seborrheic keratoses are usually located on the face, neck and trunk.

These descriptions indicate that melanoma and the above categories of benign tumors differ slightly in their physical characteristics and colors. If any automated approach is to succeed in diagnosing melanoma, a collection of these features rather than a single feature needs to be used in order to obtain a satisfactory classification of the tumor images belonging to one of these categories. Indeed, this fact is also reflected by Figure 1 and Figure 2, obtained after processing and examining 326 digital images of skin growths of the above mentioned categories. These figures show some statistical data on the distribution of percentages of tumors within each class with respect to irregularity and asymmetry. Figure 1 suggests that the irregularity index (to be explained later) alone is not sufficient in diagnosing melanoma since many benign tumors have irregularity indices which are as high as those for melanoma. Similarly, Figure 2 indicates that percent asymmetry (obtained by overlapping the two halves of a tumor along the best axis of symmetry and dividing the nonoverlapping area differences of the two halves by the total area of the tumor) also does not give a satisfactory separation between melanoma and other benign tumors.

While diagnosing skin cancer, dermatologists base their clinical diagnosis decisions on experience as well as complex inferences and extensive pathophysiological knowledge. Such experience cannot be condensed into a small set of relations, and this limits the performance of algorithmic approaches of many clinical tasks. The breadth of clinical knowledge is an obstacle to the creation of symbolic knowledge bases comprehensive enough to cope with diverse exceptions which occur in practice. Experience-based

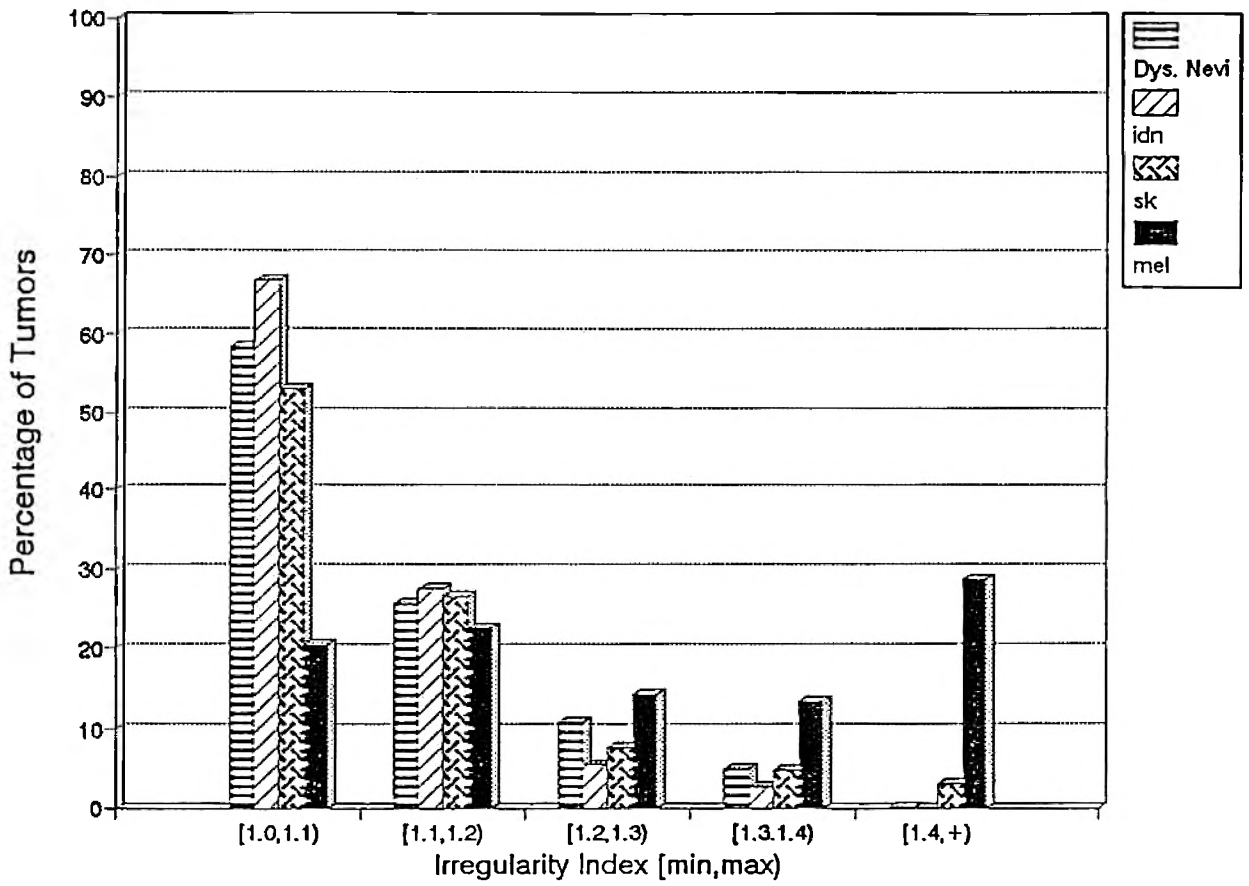

Figure 1 Irregularity Index of 326 Tumors

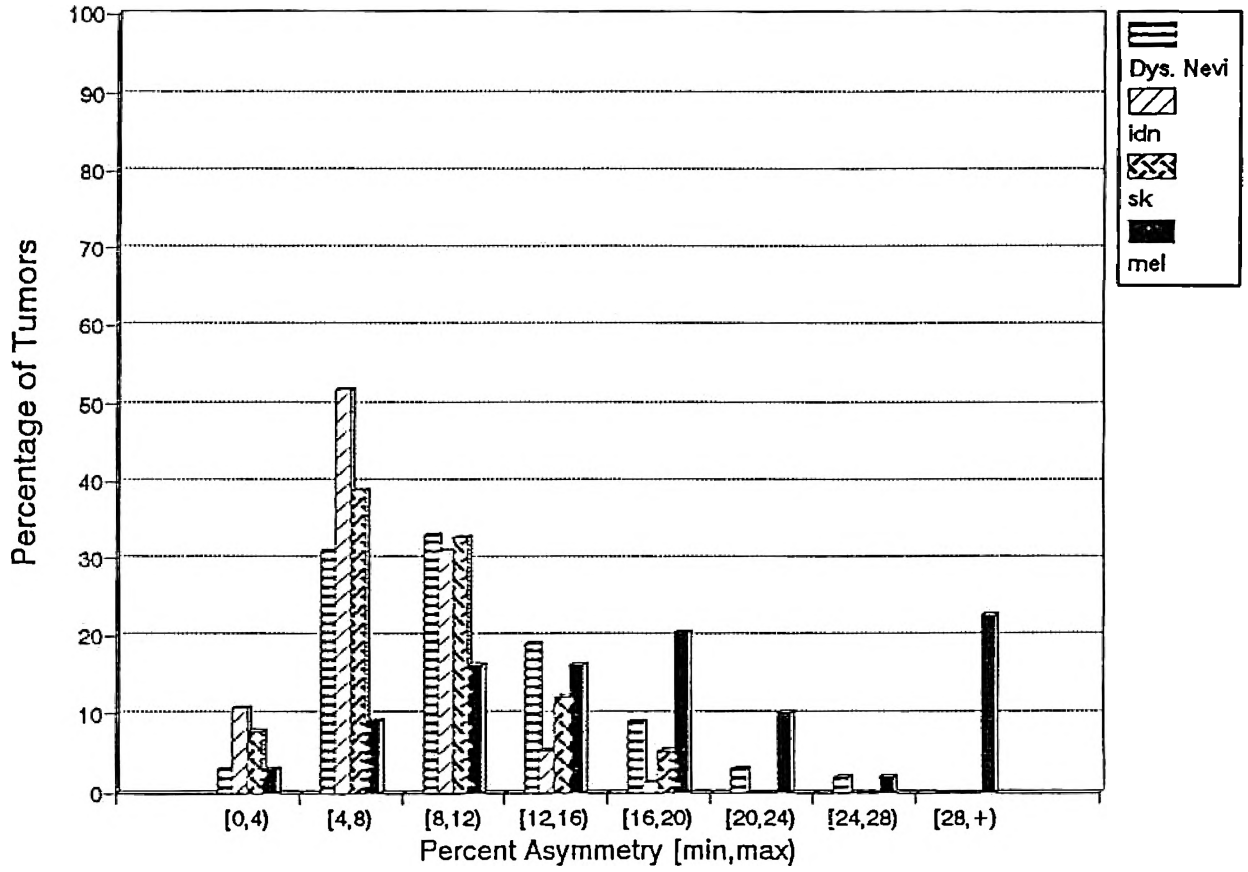

Figure 2 Percent Asymmetry of 326 Tumors

learning is the property of artificial neural networks which make them ideal for diagnostic applications such as the one above. Using the indices described above, as well as color information, a neural network should be able to learn and gain experience about the malignant melanoma diagnosis problem. The ability to select pertinent features for a particular problem on their own is an edge which neural networks possess over expert systems when solving such diagnosis problems. In the following chapter, we give a brief introduction of artificial neural networks as pattern classifiers and explain the training/testing approach for classification. In Chapter 3, we describe our approach to diagnosing the melanoma tumors and the selection and derivation of the features used for this purpose.

B. Artificial Neural Networks

In recent years, neural networks have been used as pattern classifiers in medical diagnosis [5], speech [6] and pattern recognition [7], and artificial intelligence applications. This trend has even accelerated by the availability of high speed computers with large amounts of processing power and memory. There is an increasing interest in the use of neural networks to solve a variety of problems in many areas of medicine and engineering. It is a fact that adaptive non-parametric neural-net classifiers work well for many real world problems. These classifiers frequently provide reduced error rates when compared to more conventional statistical approaches and are a powerful and flexible means for mapping a fixed number of inputs into a set of discrete classes. These characteristics make artificial neural networks a strong candidate for diagnostic problems where a set of symptoms is mapped to a set of possible diagnostic classes. In our

research, we are motivated by the desire to classify skin tumors as malignant or non-malignant from color photographic slides of the tumors and to further explore how we can add learning to this diagnosis process in order to automatically classify the skin tumors correctly.

II. NEURAL NETWORKS AS PATTERN CLASSIFIERS

Computer-based medical systems are playing an increasingly important role in assisting both diagnosis and treatment. When designing such tools, certain objectives must be considered carefully. First of all, dermatologists should be able to use low-cost, user-friendly tools such as programs running on personal computers. Nevertheless, to satisfy physicians requirements, processing time should also be short.

Since any failure of such tools could prove harmful to patients, fault tolerance and reliability are the most critical characteristics. At the same time, end users must be provided with as much information as possible about how the processing is carried out.

In the effort to reach these objectives, developers of computer aids for physicians face a variety of problems originating from the complex nature of the biological data. Such data are characterized by an intrinsic variability that can occur as the result of spontaneous internal mechanisms or as a reaction to occasional external stimuli. Furthermore, most biological events result from the interaction of many systems and subsystems whose different effects are almost indistinguishable.

Clinicians are accustomed to such problems, but their skills cannot be easily incorporated in computer programs. Most clinical decisions are based on experience as well as on complex inferences and extensive pathophysiological knowledge. Such experience cannot be condensed into a small set of relations, and this limits the performance of algorithmic approaches to many clinical tasks. The breadth of clinical

knowledge is an obstacle to the creation of knowledge bases comprehensive enough to cope with the diverse exceptions that occur in practice.

Experience-based learning, fault tolerance, graceful degradation, and signal enhancement are properties of artificial neural networks that make them effective in solving the above problems. This points to a way for implementing reliable computer-based medical systems that can closely emulate a physicians expertise.

This thesis describes a neural network system for diagnosing skin cancer. The recent resurgence of interest in neural networks, machine learning, and parallel computation has led to renewed research in the area of statistical pattern classification. Early pattern classification research performed in the 60's and 70's focussed on asymptotic (infinite training data) properties of classifiers. The thrust of recent research has changed. More attention is being paid to practical issues as pattern classification techniques are being applied to speech, vision, robotics, and artificial intelligence applications where real time response with complex real-world data is a necessity. Much of this research is motivated by the desire to understand and build parallel neural net classifiers inspired by biological neural networks and by the need to add learning to artificial intelligence applications. This has led to an emphasis on robust, adaptive, non-parametric classifiers that can be implemented on parallel hardware.

Adaptive non-parametric neural-net classifiers work well for many real world problems. These classifiers frequently provide reduced error rates when compared to more conventional Bayesian classifiers and also provide selection of differing practical characteristics. Classifiers provide trade-offs in memory, computation, training time, and

adaption requirements. They also differ in ease of real-time implementation using custom VLSI circuitry, in the ease with which they can be programmed efficiently on specific parallel or serial computers, and in computational complexity. Generalization capabilities for specific applications and the ease with which the complexity of a classifier can be matched to the amount of training data also differ. Finally, classifiers differ in their abilities to use unsupervised training data and in their ability to determine what input features contribute to the classification performance. These issues, more than error rate, tend to drive the selection of a classifier for a particular application.

A. Broad Classification of Neural Net Classifiers

Practical differences between classifiers and internal differences in how classifiers form decision regions lead to four broad groups of classifiers (Figure 3). The uppermost group of Figure 3 takes into account the most conventional or Bayesian Classifiers, while the lower three groups contain adaptive classifiers. These adaptive classifiers can all be implemented using fine grain parallelism. Most also require simple local computations for incremental adaptation and can form arbitrary decision regions.

1. Probabilistic Classifiers. Probabilistic classifiers (see Fig. 3) assume a priori probability distributions such as Gaussian or Gaussian mixture distributions for input features. Parameters of distribution are typically estimated using supervised learning where all the training data is assumed to be available simultaneously. These classifiers provide optimal performance when the underlying distributions are accurate models of the test data and sufficient training data is available to estimate distribution parameters

Group	Computing Element	Representative Classifier
Probabilistic	Distribution Dependent	Gaussian Mixture
Hyperplane	Sigmoid	Multi-Layer Perceptron, Boltzmann Machine
Receptive Fields (Kernel)	Kernel	Method of Potential Functions, CMAC
Exemplar	Euclidean Norm	K-Nearest Neighbour, LVQ

Figure 3 Four Basic Classifier Groups (from Lippman[8])

accurately. These two conditions are not often satisfied in nonstationary environments with real world data.

2. Hyperplane Classifiers. Hyperplane Classifiers form complex decision regions using nodes that form hyperplane decision boundaries in the space spanned by the inputs. Nodes typically form a weighted sum of the inputs and pass this sum through a sigmoid nonlinearity, as shown in Fig 3. Other nonlinearities, including high order polynomials of the inputs, are also used. These classifiers have low memory and computation requirements during classification but may require long training times and/or complex training algorithms. They include multi-layer perceptrons trained with back-propogation (back propogation classifiers) [8], Boltzmann machines [9], binary-tree classifiers, high order nets that form high order polynomials of inputs [10] and high order nets resulting from the use of Group Method of Data Handling (GMDH) algorithms [11].

3. Kernel Classifiers. Kernel or receptive field classifiers create complex decision regions from kernel-function nodes that form overlapping receptive fields. Kernel-function nodes use a kernel function, as shown in Fig. 3, which provides the strongest output when the input is near the nodes centroid. Some of the important properties of Kernel classifiers is that they train rapidly, can use combined supervised/unsupervised training and have intermediate memory and computation requirements. Neural net kernel classifiers include map-based approaches that use arrays of nodes which compute kernel functions, classifiers based on the Cerebral Model Articulation Controller (CMAC) [12], and classifiers that use the method of potential functions [13], often called radial basis functions.

4. Exemplar Classifiers. As the name implies the exemplar classifiers perform classification based on the identity of the training examples, or exemplars, that are nearest to the input. The nearest neighbors can be determined using exemplar nodes that are similar to the kernel-function nodes. The exemplar nodes compute the weighted Euclidean distance between inputs and node centroids. Centroids correspond to previously presented labeled training examples or to cluster centers formed during combined unsupervised/supervised training. Exemplar based classifiers train rapidly but may require large amounts of memory and computation time for classification. Exemplar classifiers include k-nearest neighbor classifiers [11], the feature map classifier [14] and Adaptive Resonance Theory (ART) classifiers [15].

B. Backpropagation Classifiers

Backpropogation classifiers have received the most attention by pattern classification researchers. This class of neural networks form nonlinear discriminant functions using single- or multi-layer perceptrons with sigmoidal nonlinearities. Backpropagation classifiers are trained with supervision, using gradient-descent training techniques which minimize the squared error between the actual outputs of the network and the desired outputs. Patterns are applied to the input nodes that have linear transfer functions. Other nodes typically have sigmoidal nonlinearities. The desired output from output nodes is low (0 or < 0.1) unless that node corresponds to the current input class, in which case it is high (1.0 or > 0.9). Each output node computes a nonlinear discriminant function that distinguishes between one class and all other classes. Early

interest in Backpropagation classifiers training was caused by the presupposition that it might be used in biological neural nets. Although this now seems unlikely, backpropagation classifiers have been successfully applied in many areas. Multi-layer perceptrons trained with backpropagation have been successfully used to:

- Classify speech sounds [16]
- Form test-to-phenome rules [17]
- Deduce the secondary structures of a protein from its aminoacid sequence
- Discriminate between underwater sonar returns [18]
- Learn good moves for backgammon [19]
- Perform nonlinear signal processing [20]

A number of theoretical analyses have been performed to determine the capabilities of classifiers based on multilayer perceptrons. Similar constructive proofs, developed independently [21] [22] demonstrated that two hidden layers are sufficient to form arbitrary decision regions using multilayer perceptrons with step function hard-limiting nonlinearities (node outputs of 0 or 1). This constructive proof was extended to suggest how multi-layer perceptrons with two hidden layers, linear output nodes, and sigmoidal nonlinearities approximate complex nonlinear functions [23]. More recent work demonstrated that multi-layer perceptrons with only one hidden layer could form complex disjoint and convex decision regions [24]. This work was followed by a careful mathematical proof [25], which demonstrates that continuous nonlinear mappings can be closely approximated by multi-layer perceptrons with only one hidden layer. This proof implies that arbitrary decision regions can also be approximated by

multi-layered perceptrons with only one hidden layer. This proof however is not constructive and does not indicate how many neurons are required in the hidden layer. Other recent theoretical work has demonstrated the advantages of sigmoidal nonlinearities over linear nodes for single-layer perceptrons trained with backpropagation. One major characteristic of backpropagation classifiers is long training times. Training times are typically longer when complex decision regions are required and when the networks have more hidden layers. As with other classifiers, training time is reduced and performance is improved if the size of the network is tailored to be large enough to solve a problem but not so large that too many parameters must be estimated with limited training data. Other techniques that have been effective in reducing training time with speech data are to update weights after presenting each training example instead of after cycling through all the examples, to randomize the presentation order of training examples, and to normalize components of input training vectors to have mean values of zero [26]. Other characteristics of back propagation classifiers that may be difficult to alter include difficulty in interpreting and understanding network solutions, and the frequent necessity of many nodes and connection weights. Research on developing techniques to design minimal-size backpropagation classifiers [27] and to develop analysis techniques to interpret the solutions found by backpropagation classifiers suggests approaches to these issues. Shorter training times and these other characteristics can, however, be obtained using other classifiers that can be implemented using fine-grain parallelism.

C. Training and Testing

The goal of pattern classification is to assign input patterns to one of a finite number, M, of classes. In the following section it will be assumed that input patterns consist of static input vectors x containing N elements or continuous valued real numbers denoted $x_1, x_2, ..., x_N$. Elements represent measurements of features selected to be useful for distinguishing between classes. Input patterns can be viewed as points in the multidimensional space defined by the input feature measurements. The purpose of a pattern classifier is to partition this multidimensional space into decision regions that indicate to which class any input belongs. Conventional Bayesian classifiers characterize classes by their probability density functions on the input features from these densities. Adaptive non-parametric classifiers do not estimate probability density functions directly but use discriminant functions to form decision regions.

The application of a pattern classifier first requires selection of features that must be tailored separately for each problem domain. Features should contain information required between classes, be insensitive to irrelevant variability in the input, and also be limited in number to permit efficient computation of discriminant functions and to limit the amount of training data required. Good classification performance requires selection of effective features and also selection of a classifier that can make good use of those features with limited training data, memory and computing power. Following feature selection, classifier development requires collection of training and test data, and separate training and test or use phases. During the training phase, a limited amount of training data and an a priori knowledge concerning the problem domain is used to adjust

parameters and/or learn the structure of the classifier. During the test phase, the classifier designed during the training phase is evaluated on new test data by providing classification decision on each input pattern. Classifier parameters and/or structure may then be adapted to take advantage of new training data or to compensate for nonstationary inputs, variation in internal components, or internal faults. Further evaluations require new test data.

The training/test set paradigm is used extensively in statistical studies. This paradigm, simply stated, consists of separating the data or samples into two distinct sets. One set is used for training, or during the learning phase of the network, and the other set is used for testing the network. These two sets should be statistically independent to allow unbiased results to be obtained on the test set. In order to generate the best classification network possible, the size of the training set should be maximized, but in order to have high levels of confidence in the results as an estimate of future performance, the size of the test set should also be maximized. This dilemma leads many researchers to arbitrarily use 50 percent of the set for training and 50 percent for testing. For this research, results are reported with various sizes of training and testing sets. This method provides more complete information than would be obtained with a fixed set size and allows for observation of trends in the data.

III. SELECTION OF FEATURES FOR DIAGNOSIS

A. Selection of Features for Diagnosing Melanoma

Diagnosis applications require a selection of features that must be tailored separately for each problem domain. The features selected should contain enough information to distinguish between classes as well as being insensitive to irrelevant variability in the inputs. On the other hand, the features must be limited in number for two reasons: 1) To keep the training (learning) time within reasonable limits, and 2) to allow the network to compute the discriminant functions efficiently with a small size training set. As a result of our analysis of the diagnosis problem, we have defined 14 features that we believe to be well discriminative between images belonging to malignant melanoma and the three benign tumors of interest here. This chapter provides a description of the selected features as well as the methodology used to extract them from the color skin images.

1. Boundary Detection. Boundary detection of skin tumors is one of the first steps (low level processing) to be performed in skin cancer recognition. All of the 14 features that were identified to be useful in the diagnosis of skin cancer required detection of the border of the tumor in the color image. The algorithm used here is an enhanced version of the radial search algorithm which was proposed in an earlier study [28]. Instead of detecting individual border points, the new method detects connected tumor segments

from which border points are determined [29]. This technique eliminates most of the spurious border points due to noise. Briefly, the border finder uses the following steps:

a. Image Smoothing and Enhancement. A median filtering algorithm is applied repeatedly to smooth the image and diminish spurious effects that may be present due to noise. The advantage of median filtering is threefold: it preserves edge sharpness of tumors, diminishes flash areas and enhances tumor contrast over the background while eliminating noise.

b. Segmentation. Image pixel values are transformed into a new plane to allow easy separation of the tumor and skin pixel values and thresholding is applied to segment the image into two distinct areas; tumor and the background (skin).

c. Border Determination. First, the tumor portion is separated from the segmented image by using a region growing algorithm and masking all the unnecessary information around it, then a ray probing algorithm is used to identify the boundary points. These points are connected by a cubic-spline to get a smooth outline of the border.

B. Feature Selection

After the boundary of the tumor area is determined, the next step is to compute the indices corresponding to each feature needed for diagnosis. In this section, we describe those features of interest and how to compute them.

1. Irregularity Index. Malignant melanoma is characterized by the irregularity in its tumor border. Irregularity is measured by an index (I) :

$$I = \frac{P^2}{4\pi A} \qquad (1)$$

where, P = perimeter of the tumor in pixels, A = area of the tumor in pixels.

The irregularity index for a circle is one (perfectly regular). In our research the perimeter and area are computed in terms of pixel counts. Figure 1 shows the irregularity index for different categories of tumors. It is clear that most melanomas have a high irregularity index, i.e., they have an irregular shape. However, there is a significant percentage of other tumors with high irregularity indices. Hence, this feature alone is not sufficient enough to discriminate melanoma from other benign types of tumors.

2. Percent Asymmetry. Asymmetry is another characteristic of malignant melanoma. Asymmetry is computed by finding an axis that is closest to the axis of symmetry of the tumor (i.e. the axis around which, if the tumor is folded into half, there is maximum overlap of the two halves). Then percent asymmetry is computed by overlapping the two halves of a tumor along the best axis of symmetry and dividing the nonoverlapping area differences of the two halves by the total area of the tumor. As we observe in Figure 2, 88.4% of the melanomas in our database of images have an asymmetry percentage above 8 percent, whereas this figure is 66%, 50.3% and 37.7% for

the dysplastic nevi, sk and idn respectively. Again, this index alone is not powerful enough to discriminate malignant melanoma from other tumors but together with other features it is expected to play a very important role in the diagnosis of melanoma.

3. Color Features. One of the most predictive features in identification of malignant melanoma is variegated coloring(VC) [30]. Dermatologists define variegated coloring as the swirling together of tan, brown, red and black giving the tumor a varied coloring. Such variegation in color implies a high variance in red (R), green (G), and blue (B) color components. Therefore, out of 12 color features, three of them are selected to be the variances in the R, G, and B color planes. Since dysplastic nevi may also turn into melanoma they also have high variances in these planes but the other benign tumors have lower variances in the RGB planes (they do not exhibit variegated coloring). In addition to variances, relative chromaticity of tumors (in RGB planes) are also added to the feature list since these features are important in discriminating melanoma from sk and idn. The relative chromaticity is defined as the normalized value of that color in the tumor area subtracted from the normalized value for the color in the background.

For example the relative chromaticity of red is defined as:

$$R_r = \frac{R_{fg}}{R_{fg}+B_{fg}+G_{fg}} - \frac{R_{bg}}{R_{bg}+B_{bg}+G_{bg}} \qquad (2)$$

where r_{fg}, g_{fg} and b_{fg} denote tumor RGB components and r_{bg}, g_{bg} and b_{bg} denote background RGB components. The relative color was defined as the color difference vector, i.e. difference in the color space between tumor and the background, or normal flesh. Reasons behind the development of a relative color concept are stated in [30] as follows: 1) to equalize any variations caused by lighting, photography/printing, or digitization process, 2) to equalize variations in normal skin color between individuals, and 3) the human visual system works on a relative color system.

Previous studies in diagnosing melanoma [30] with an expert system indicate that spherical color space coordinates gave better diagnosis results than the RGB, CIE or IHS color spaces. Therefore, we also added these indices into our set of input features. The equations to transform from (R, G, B) to spherical coordinates are given by [30]:

$$L=\sqrt{R^2+G^2+B^2} \tag{3}$$

$$AngleA=\cos^{-1}\left[\frac{B}{L}\right] \tag{4}$$

$$AngleB=\cos^{-1}\left[\frac{R}{L\times\sin(AngleA)}\right] \tag{5}$$

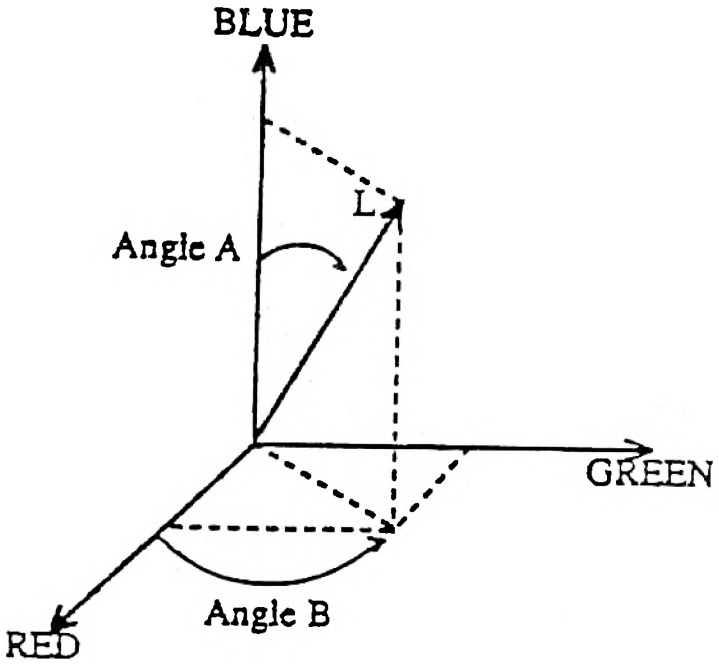

Figure 4 The Spherical Transform (from [30])

This transformation splits the color space into a two-dimensional color space, represented by two angles, Angle A and Angle B; and a one dimensional intensity (brightness) space, represented by the vector length L (see Fig. 4). To compute Length, Angle A and Angle B for each tumor image, we found Length, Angle A and Angle B for each of the pixels in the tumor and took an average of them.

From the viewpoint of color clustering, it is desired that the image be represented by color features which constitute a space possessing uniform characteristics such as the (L*, a*, b*) color coordinate system [31]. Since sk's and idn's are brighter in color (closer to white) than melanoma and dysplastic nevi, they have distinct values in this color space. Dermatology imaging researchers also believe that this space may be useful in distinguishing melanoma from dysplastic nevi due to small differences in lightness, hue and chroma between dysplastic nevi and melanoma (according to dermatologists dysplastic nevi are brighter and have less blue, i.e. more relative red components). In our research, lightness, hue and chroma are computed for each point in the tumor using the formulas given in [31] and then an average is taken for all the pixels in the tumor.

IV. NEURAL NETWORK DESIGN AND EXPERIMENTAL RESULTS

A. Diagnosis of Malignant Tumors Using a Neural Network

For the research reported here, the discriminant features explained above were extracted from 210 digital images of skin cancer. These images were obtained from the clinical collection of Dr. William Van Stoecker (see Appendix B), a resident dermatologist in Rolla, MO, and some of the images were obtained from New York University Medical School. All these images were 512 x 512 pixel color images with 24 bits per pixel (8 bits for each R, G and B planes). Ninety-six images were in the malignant melanoma category and there were 111 images of dysplastic nevi (dys nevi), 58 intradermal nevi (idn) and 61 seborrheic keratoses (sk).

B. Neural Network Implementation

A feedforward artificial neural network with 14 inputs (see Table I) and one output (indicating whether the tumor is malignant melanoma or not) was used and trained using the backpropagation rule. A versatile neural network development software package, NeuralWorks Professional, was used for the experiments which follow and a customized version of this Neural Network was also implemented. Details of the implementation are found in Chapter 5.

One major characteristic of backpropagation classifiers is long training times. Training times are typically longer when complex decision regions are required and when

Table I. Features Used For Diagnosis

Feature Description	**Number of Inputs**
Irregularity	1
Asymmetry	1
Variance in the RGB Plan	3
Relative Chromaticity	3
Spherical Transform	3
(L^*, a^*, b^*) Color space	3

networks have more hidden layers. One way of solving this problem is to use few hidden layers.

In this study, only one hidden layer was used based on the fact that it performed reasonably well among several network configurations with which we had experimented and also produced fast results. Typical training times varied between 40-60 minutes. Another technique that we have used in reducing the training time is randomization of the presentation of the order of the training examples by using a "shuffle and deal" randomization scheme. Other techniques which are effective in reducing training time with some applications are to update weights after presenting each training example instead of after cycling through all the examples.

Training of the network was continued with several epochs of the training set until the root mean square error of the output was below 0.05. Testing was done and the success rates for the correct diagnosis of melanoma as melanoma and non-melanoma as non-melanoma were recorded. Results were obtained for training/testing percentages of 20/80, 40/60, 60/40 and 80/20.

C. Experimental Design and Test Results

The experiments were designed to test the effectiveness f the input features in discriminating the melanoma images from the others. Three sets of experiments were conducted, each repeated twice; once with dysplastic nevi included and another time with dysplastic nevi excluded, resulting in a total of six experiments. The reason for repeating the same experiments without dysplastic nevi is the fact that dysplastic nevi are precursors of melanoma and they possess the same variegation of coloring as melanoma tumors. By eliminating dysplastic nevi, classification is expected to become easier for the network. Those experiments with dysplastic nevi included used a total of 210 images (96 melanomas, 43 dysplastic nevi, 30 idn, and 41 sk) while those with dysplastic nevi excluded used 216 images (96 melanomas, 58 idn, and 62 sk) for training plus testing.

The primary focus for training was to be able to distinguish melanoma from benign tumors. Experimentation has shown that the total number of melanomas in the training set needs to be close to 50% of the whole population in order to obtain good diagnostic results.

The reason for the varying numbers used for each class is that we tried to maximally utilize the images available in the database for training and testing while, at the same time, keeping a good balance of different types of tumor images. For both experiments, 96 melanoma images were used and the total number of non-melanoma images were kept within a margin not exceeding 56% of the whole population. When X% of images was used for training, the remaining images (100-X%) were used for testing.

Each class contributed the same percentage of their total number to the training and test sets.

Experiments differed from one another by the set of input features used. The first set of experiments was conducted with the 14 input features originally described.

The second set of experiments was designed to test the effects of the use of different film types in the diagnosis process. In these experiments, to offset the effects of the different films used, spherical color space coordinates and (L*, a*, b*) color coordinates were removed from the input set leaving only those color features related to the relative color concept (color variances and the relative color). Hence, only eight input features were used in this phase.

In the third set of experiments, two new features were experimented with. The first of these was elevation and the second was area. These measures were determined in a subjective fashion by a dermatologist. The dermatologist classified tumors as having a marked elevation or no elevation. Also the dermatologist determined whether the area of the tumor was greater than 6mm from color slides of the tumor. These features were then incorporated into the image database in the form of binary vectors.

The results of these experiments are summarized and plotted in the following paragraphs.

1. Experiments 1a and 1b. Experiment 1a was conducted with all four classes, melanoma, idn, dys nevi, and sk, while dysplastic nevi images were removed from Experiment 1b. A total of 210 and 216 images were used altogether for training plus

testing for experiments 1a and 1b, respectively, with 14 input features supplied per image. Results are plotted in Figures 5a and 5b.

In Experiment 1a, for training percentages exceeding 60%, melanomas are diagnosed with close to 90% success rate. The sks and idn's are always above 90 percent for training percentages of 40%or above (see Fig. 5a). The dysplastic nevi are however quite inconsistent and vary between a low of 50 and a high of 85 percent. We believe this is due to the fact that dysplastic nevi are precursors of melanoma and they possess the same variegation of coloring as melanoma tumors.

In Experiment 1b, the results improve appreciably (Fig. 5b) for melanoma with successful diagnosis rate not below 92% for any case, peaking at 96%. The other two categories did not exhibit any significant changes and were diagnosed with success rates of 100% for training sizes above 60 percent, with the exception of idn showing a poor performance for the training percentages of 40% or below. This result supports the original observation that dysplastic nevi are precursors of melanoma and they possess the same variegation of coloring as melanoma tumors. Hence, elimination of the dysplastic nevi images from the training set made the classification job easier for the network and the number of false negatives were reduced considerably. As a result, the overall performance (the curve with a solid black icon) was boosted considerably (to a 98% success rate with a training set size of 80%).

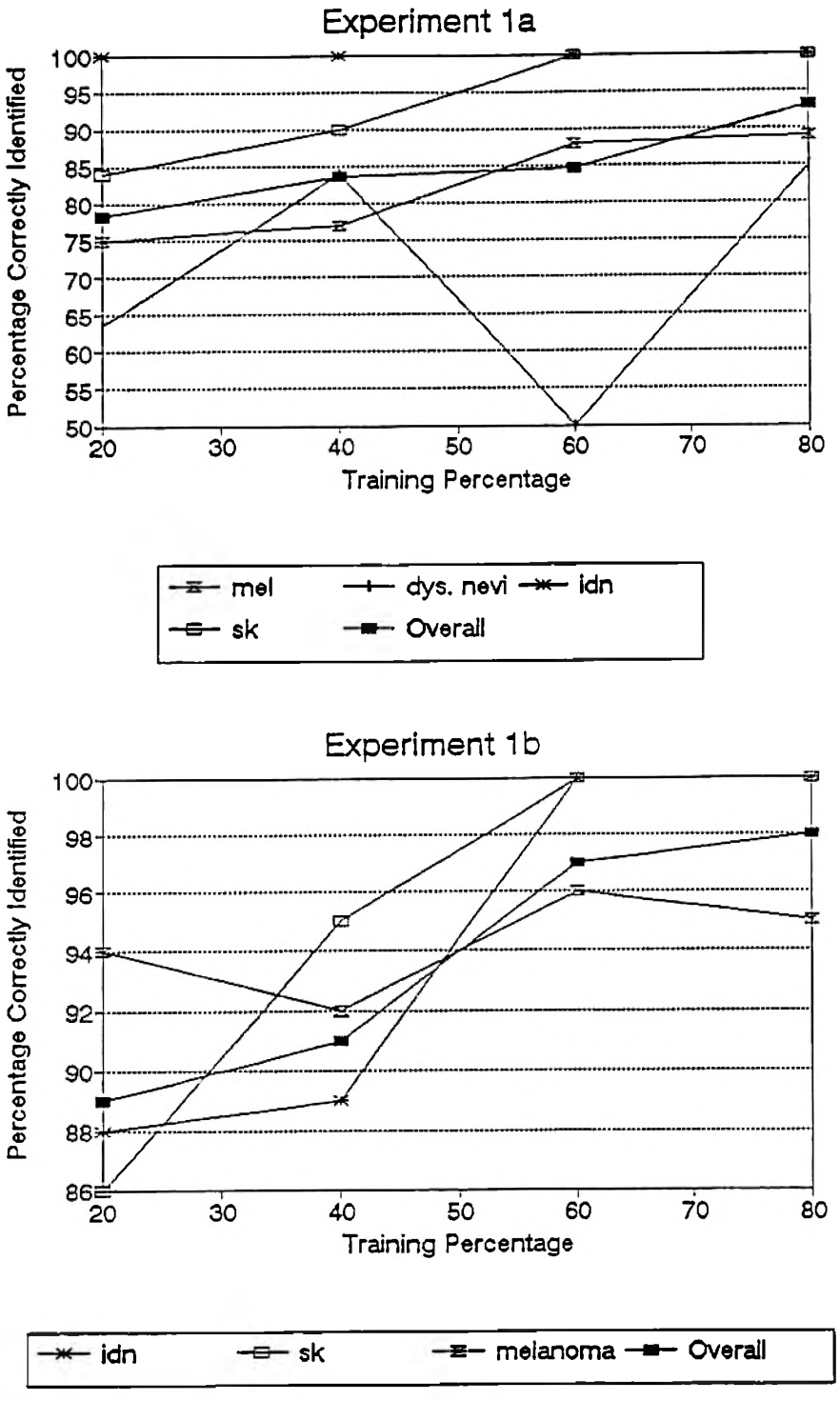

Figure 5 a) Success Rate for Experiment 1a; b) Success Rate for Experiment 1b.

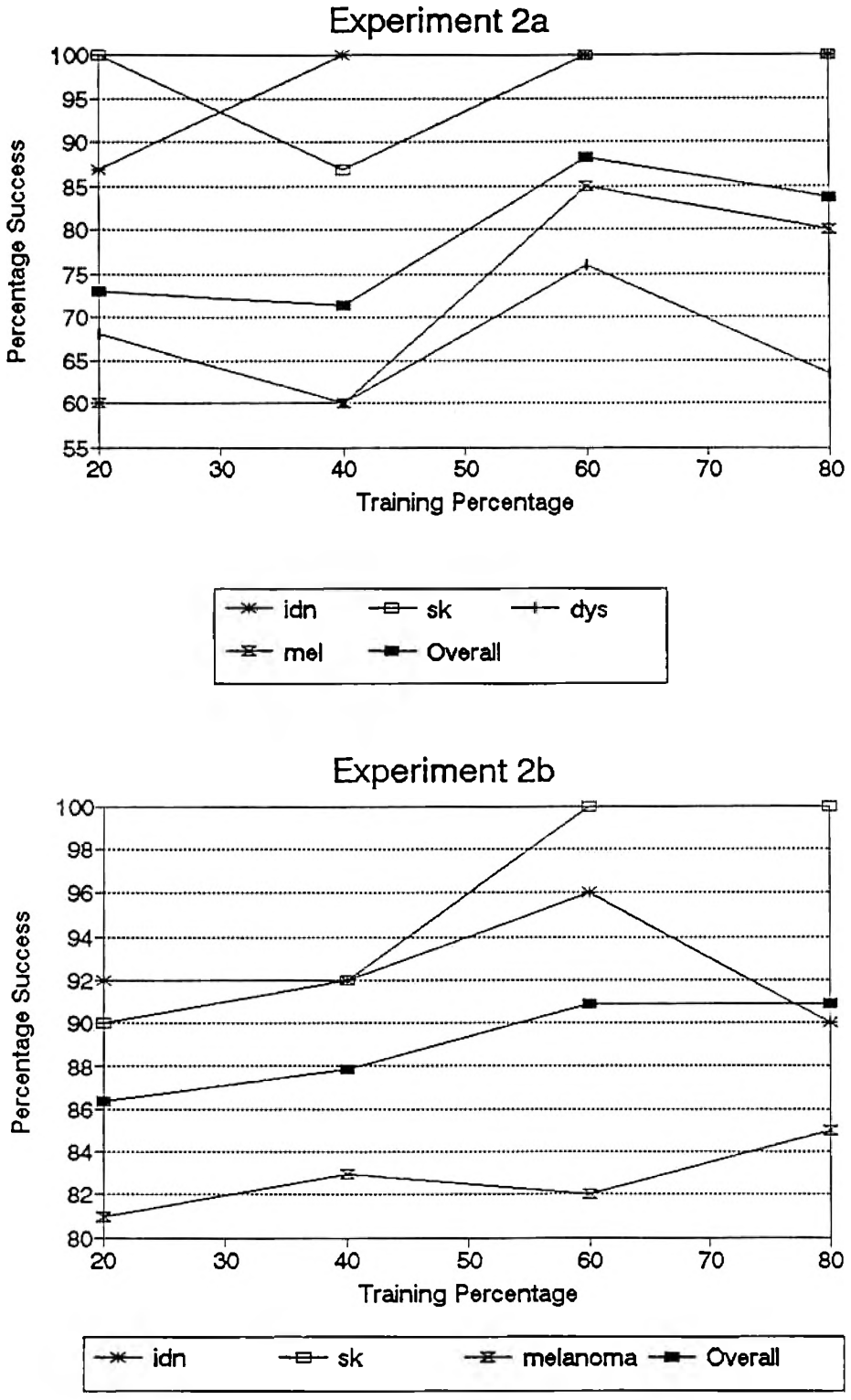

Figure 6 a) Success Rate for Experiment 2a; b) Success Rate for Experiment 2b.

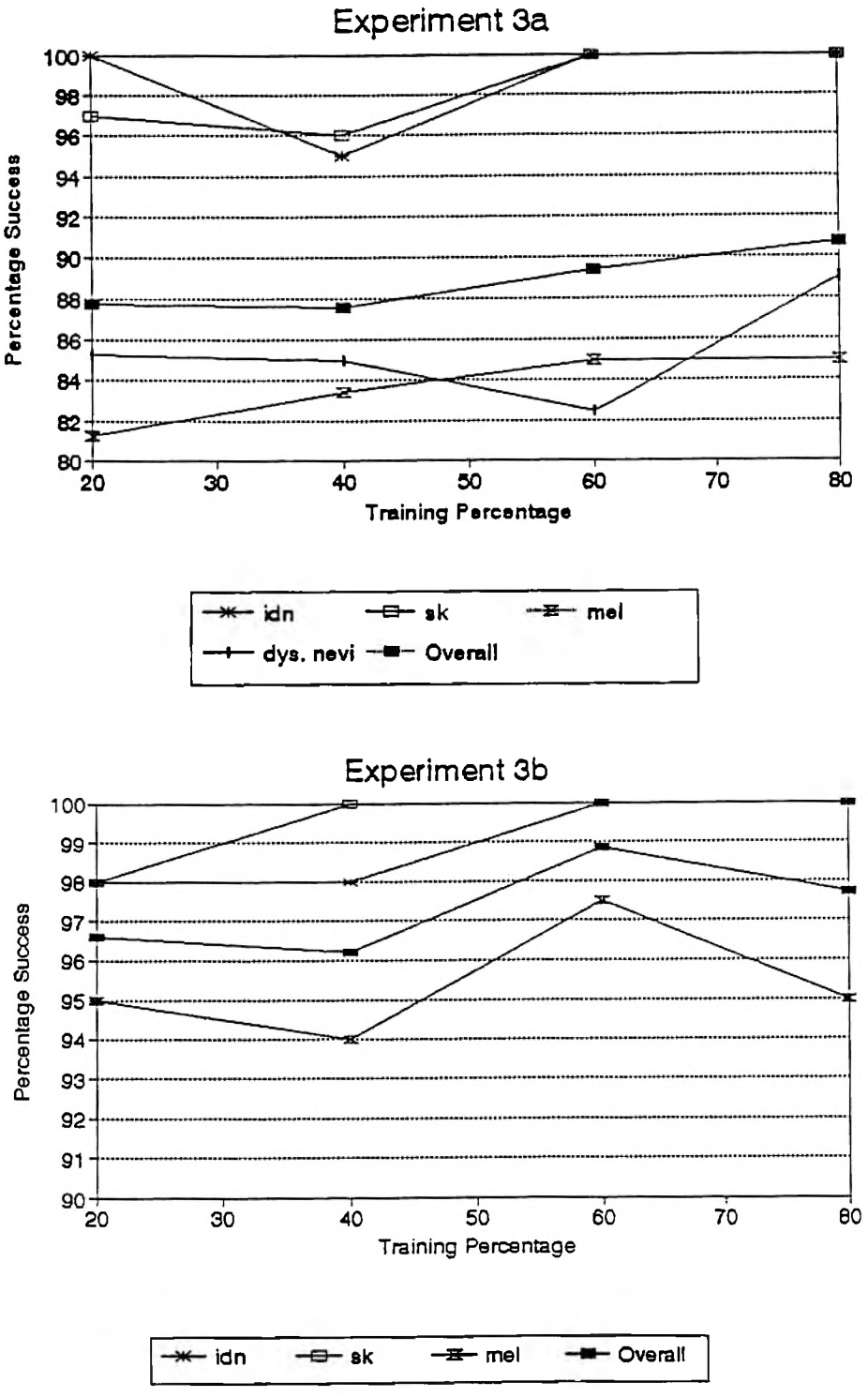

Figure 7 a) Success Rate for Experiment 3a; b) Success Rate for Experiment 3b.

2. Experiments 2a and 2b. The same procedure used for experiment 1 was repeated for this set of experiments except that 8 input features were used instead of 14. The goal was to test the effect of the types of film used. In our image database, all the melanoma and dysplastic nevi slides were Ektachrome while a majority of the sk and idn slides were Kodachrome. To offset the effect of the different film types used, absolute color components in the input, namely spherical color space coordinates and (L*, a*, b*) color coordinates, were removed from the input set leaving only those color features related to the relative color concept (color variances and the relative color). Hence, only eight input features were used in this phase.

Obviously, elimination of all the absolute color information from the input is expected to cause the success rate to go down due to the degradation of the discriminant features in the input. However, we should not expect a significant change from the previous results, which would otherwise be interpreted as due to the film type. As illustrated by the plots in Figures 6a and 6b, the change in corresponding success rates is not large enough to raise concerns about the effect of film type on the results.

However, it was observed that the melanoma success percentages in Exp. 2b were relatively lower than those of Experiment 1b (10% drop). This result can be explained due to the importance of absolute color information in the input. Absolute color information is important in the diagnostic process particularly from the viewpoint of color clustering (shades of tan, brown and black, dashes of red, white and blue are signs of malignancy) and brightness information of tumors in the form of the brightness vector in the spherical

transform domain. Hence, including as much color information about tumors as possible helps the neural network in diagnosis of the malignant and nonmalignant tumors.

3. Experiments 3a and 3b.

The aim of experiments 3a and 3b is to experiment with the inclusion of two more indices to the neural network. These are area of the tumor and elevation. Dermatologists believe that there is a weak correlation between the elevation of a tumor and nonmalignancy. This belief is inspired from the fact that few malignant tumors have a marked elevation whereas many categories of nonmalignant tumors e.g. basal cell carcinoma, sebhoerric keratoses and intradermal nevi are characterized by crust (as noted in their descriptions). Also area is important according to dermatologists. Since melanoma is an uncontrolled growth of cells, dermatologists believe that on the average melanoma tumors are likely to be larger in size than nonmalignant tumors in the same stage. A group of experiments to test the validity of these two hypotheses were performed. In one case area was added to the 14 features. However the success rates of this network were not as good as without the area index. The primary reason for the lower performance of this network is the fact that many of the malignant melanoma tumors in our database are in their incipient stages (since patients report to a dermatologist in the early existence of the tumor). Hence even though the area index may be important, our database of images does not reflect this fact and hence experiments with area index did not yield satisfactory results.

Hence Experiments 3a and 3b were performed with 14 features plus an elevation index. This index was determined by a dermatologist after examining all the tumor slides and noting if there was a marked elevation (greater than 1mm) on the tumor or not.

The success rates for dysplastic nevi (see Fig. 7a) went up in experiment 3a. This is probably due to the fact that many of the dysplastic nevi have some elevation whereas most malignant tumors like melanoma do not. The success rates for the other categories of nonmalignant tumors, i.e., sk and idn, are almost perfect although melanoma success rates remain around 85 percent. This might be explained due to the fact that many of the melanoma tumors are in their incipient stages and hence do not have enough color variegation to distinguish them from dysplastic nevi. The other categories of nonmalignant tumors, i.e., sk and idn, are generally characterized by marked elevation and hence the elevation helps to increase their percentage of successful diagnosis.

Highest success rates for melanoma (see Fig. 7b) were achieved in Experiment 3b. This is due to a combination of the facts that dysplastic nevus are not present in this experiment and the fact that malignant tumors such as melanoma do not usually have crusts or elevation which characterize many nonmalignant tumors, i.e., sk and idn.

V. SIMULATION OF THE HYPERPLANE CLASSIFIER

In this chapter, the design of a back propagation simulator is discussed. In a (Fig. 8) backpropagation network, signals flow bidirectionally, but in only one direction at a time. During training, there are two types of signals present in the network: during the first half-cycle, modulated signals flow from input to output; during the second half cycle, error signals flow from output layer to input layer. See Figure 9 for an example of a hypothetical surface in weight space (with two weights). In the production mode only the feedforward, modulated output signal is utilized.

Several assumptions have been incorporated into the design of this simulator. First the output function on all hidden and output layer units is assumed to be the sigmoid function. In addition a momentum term is included in the weight-update calculations. These assumptions imply the need to store weight at one iteration, for use in the next iteration. Finally a bias term has been included in the calculation. In this network model, the input units are fan-out processors only. That is, the units in the input layer perform no data conversion on the network input pattern. They simply act to hold the components of the input vector within the network structure. Thus the training process begins when an externally provided input pattern is applied to the neurons in the input layer. Forward signal propagation occurs according to the following sequence of activities:

1. Locate the first processing unit in the layer immediately above the current layer.
2. Set the current input total to zero.
3. Compute the product of the first input connection weight and the output from the transmitting unit.

4. Add the product to the cumulative total.

5. Repeat steps 3 and 4 for each input connection.

6. Compute the output value for this unit by applying the sigmoid function

$$f(x) = \frac{1}{1+e^{-x}} \tag{6}$$

where x = input total.

7. Repeat steps 2 through 6 for each unit in the layer.

8. Repeat steps 1 through 7 for each layer in the network.

Once an output value has been calculated for every unit in the network, the values for the units in the output layer are compared to the desired output pattern, element by element. At each output unit, an error value is calculated. These error terms are then fed back to all other units in the network structure through the following sequence of steps (see Fig. 10):

1. Locate the first processing unit in the layer immediately below the output layer.

2. Set the current error total to zero.

3. Compute the product of the first output connection weight and the error provided by the unit of the upper layer.

4. Add the product to the cumulative error.

5. Repeat steps 3 and 4 for each output connection.

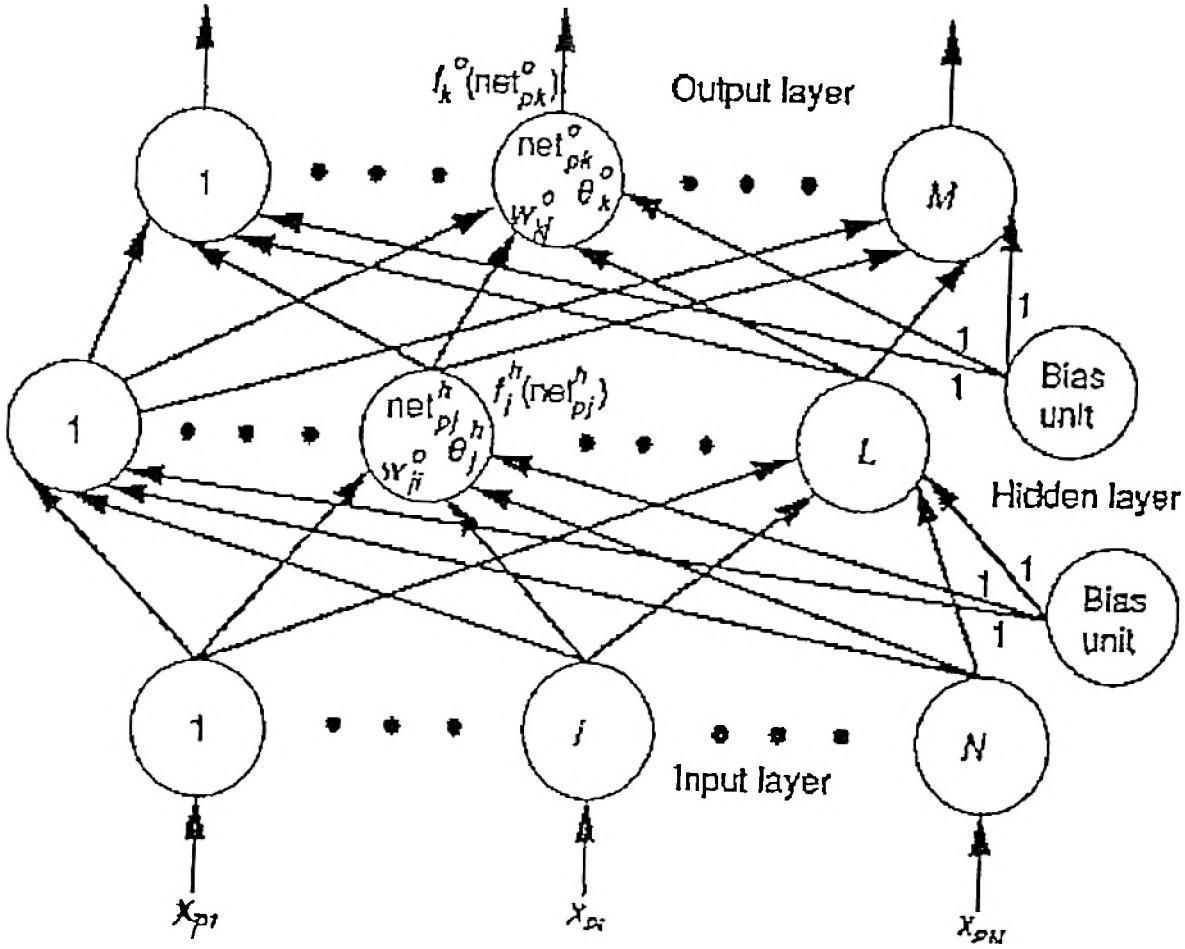

Figure 8 A Multilayer Perceptron

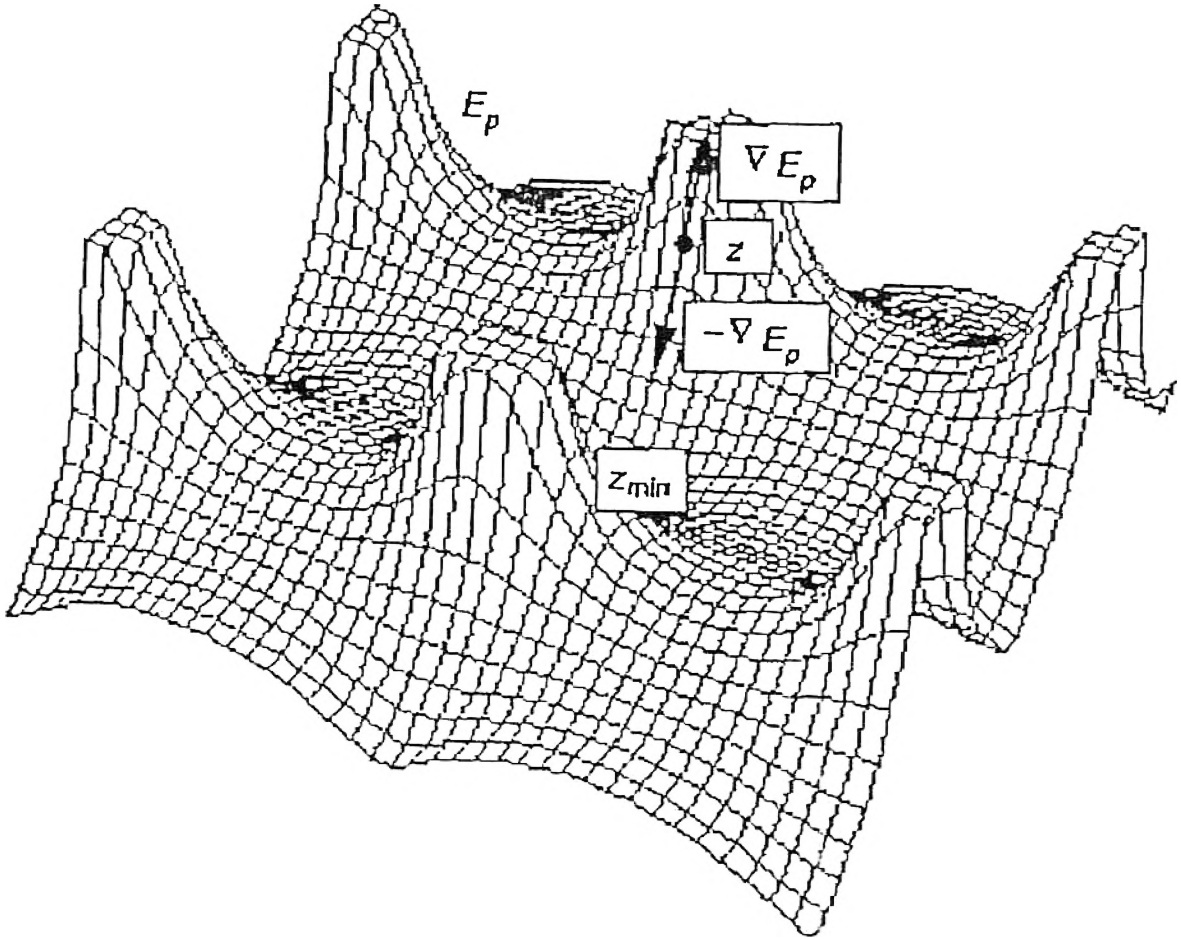

Figure 9 Hypothetical Surface in Weight Space (Simple Example of a network with two weights)

6. Multiply the cumulative error by:

$$o\ (1-o) \tag{7}$$

where o is the output value of the hidden layer unit produced during the feedforward operation.

7. Repeat steps 2 through 6 for each unit in this layer.

8. Repeat steps 1 through 7 for each layer.

9. Locate the first processing unit in the layer above the input layer.

10. Compute the weight change value for the first input connection to this unit by adding a fraction of the cumulative error at this unit to the input value to this unit.

11. Modify the weight changes term by term by adding a momentum term equal to a fraction of the weight change value from the previous iteration.

12. Save the new weight change value as the old weight change value for this connection.

13. Change the connection weight by adding the new weight change value to the old connection weight.

14. Repeat steps 10 through 13 for each input connection to this unit.

15. Repeat steps 10 through 14 for each unit in this layer.

16. Repeat steps 10 through 15 for each layer in the network.

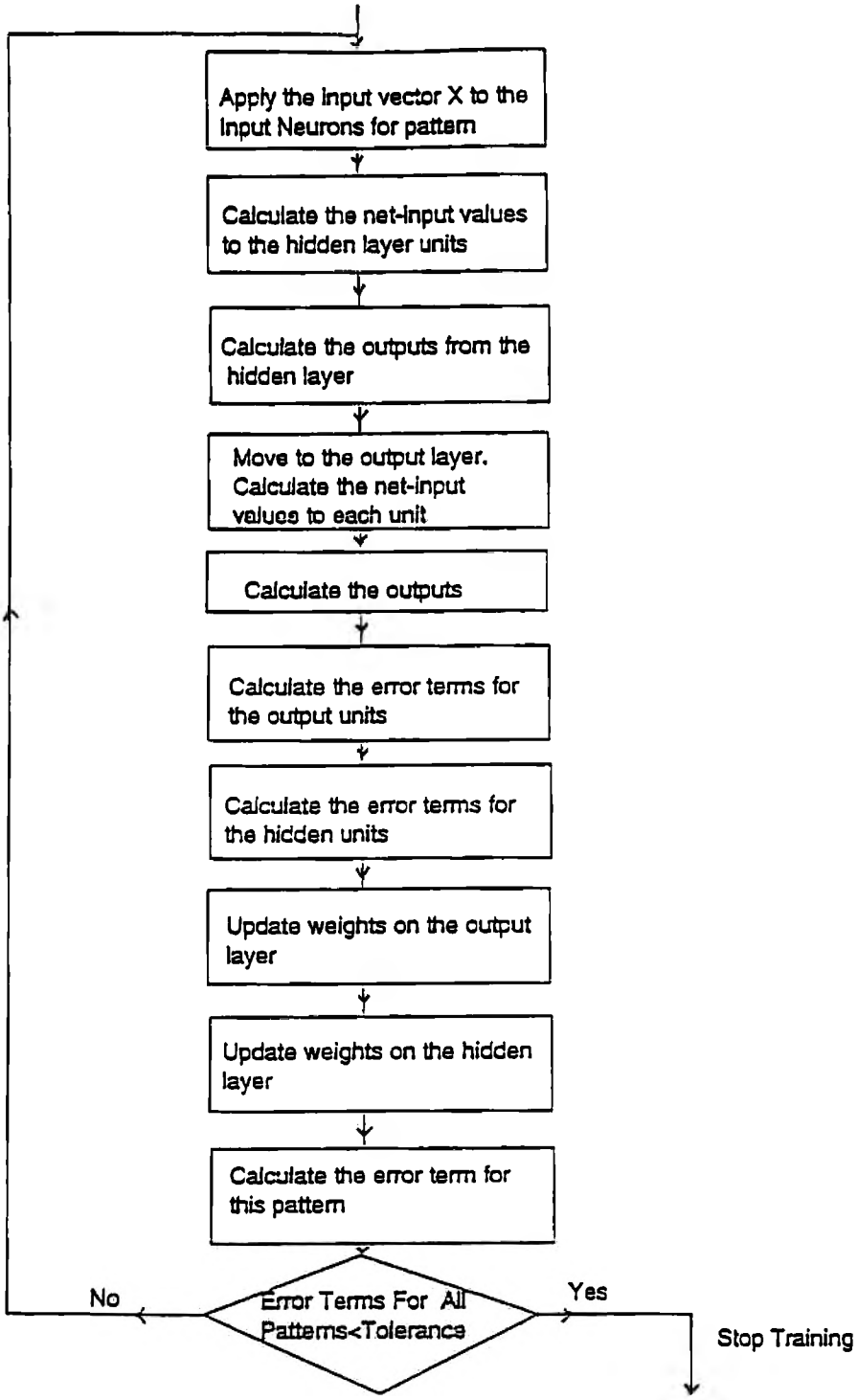

Figure 10 Flowchart of the Training Algorithm

The algorithm outlined above was implemented in AT&T C++ on a Sun Sparcstation as well as on a 486 personal computer in the Borland C++ environment. The customized software had the option of setting the number of hidden neurons, the learning rate, momentum term and the error tolerance. Training and test sets were formed in the percentages mentioned randomly so the training/test paradigm was followed. Experimentation revealed that one hidden layer with 8 hidden neurons gave the best training results in terms of time and successful diagnosis. A learning rate of 0.1 and a momentum term of 0.01 were used in all the experiments to speed up training. The root mean square error tolerance was set to 0.05. The code is listed in Appendix A.

VI. CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS

A. Conclusions from Experimental Results

A fast and effective method to separate malignant melanoma from other types of benign tumors is becoming increasingly needed due to the fact that malignant melanoma incidence has risen dramatically in recent years and early detection can save thousands of lives each year. In this study, we attempted to diagnose melanoma from color skin images using an artificial neural network. For this purpose, a set of features to distinguish melanoma from three types of benign tumors was defined and methods to measure these features from digitized color slides were described. Overall, diagnostic results were found to be very promising and as high as 97% accuracy in detecting malignant melanoma is achieved using training data sets of reasonable size (see Experiment 3b). As a result of this study, the following results are confirmed experimentally:

a) Color characteristics of tumors play a crucial role in the diagnosis process, b) tumor asymmetry and border irregularity are two important diagnostic features for distinguishing malignant melanoma from benign tumors such as seborrheic keratoses, dysplastic nevi, and intradermal nevi, c) malignant melanoma and dysplastic nevi images exhibit some similarities and therefore testing for tumor malignancy in the absence of dysplastic nevi images gives better diagnostic results. This is confirmed by the second part (b) of each experiment.

B. Suggestions for future research

From the dermatology standpoint, most common skin tumors can be put into five major groups which exhibit different characteristics regarding the need for biopsy, malignancy and first choice Rx paths (see Figure 11). These groups are: malignant melanoma (mm), dysplastic nevi (dys), squamous cell carcinoma (scc) or basal cell carcinoma (bcc), actinic keratoses (ak), and common benign lesions (intradermal nevus (idn), compound nevus (cpdn), and seborrheic keratoses (sk)).

The studies undertaken in this thesis aimed to diagnose only malignant melanoma among four types of tumors. In future research, we plan to use a hierarchical, diagnostic-tree based approach to diagnose the above five classes of skin tumors (see Fig. 12). By breaking down the problem into well-defined smaller problems and, hence, limiting the number of diagnostic outcomes to only (Yes/No) type decisions, the complexity of the whole process is reduced considerably while the diagnostic power is increased proportionally. The decision at each branch of the tree can be made by a separate neural network specifically designed and trained for classifying the tumors at that particular level of the tree. The hierarchical design in Fig. 12 indicates that four different neural networks are needed for this purpose.

Note that there will be an extra development overhead due to the implementation of multiple neural networks specialized to distinguish different classes. Each will require different input feature sets to be used. So far, we have identified overall 20 features that can be useful in any diagnostic process. In addition to the 14 features listed in Table 1, elevation, texture, 2D Fourier transform coefficients, semi-translucency, ulcer, and area

are also believed to be relevant features in the diagnostic process. We plan to statistically analyze image feature sets corresponding to each class and obtain their distributions in order to find the optimum input feature set for each neural network. For example, the experiments in this thesis verify that the 14 features listed in Table 1 are crucial in diagnosing malignant melanoma. It is important to note that a neural network should not be overloaded with extra inputs (features) which do not carry any discriminative information, i.e., which do not help in classification.

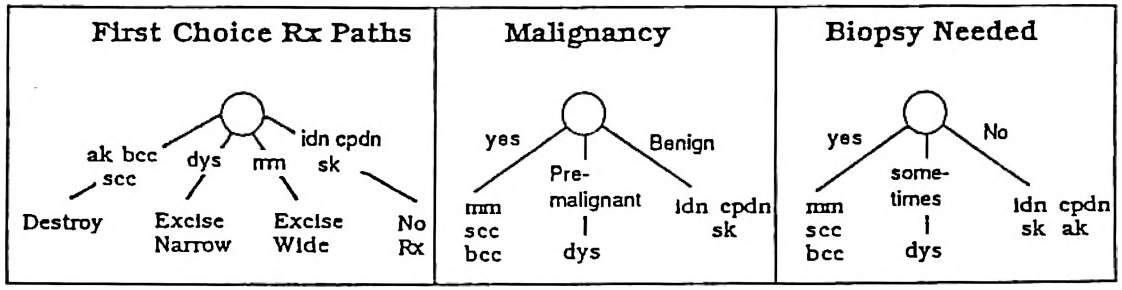

Figure 11 Outcome Tree for Most Common Skin Tumors

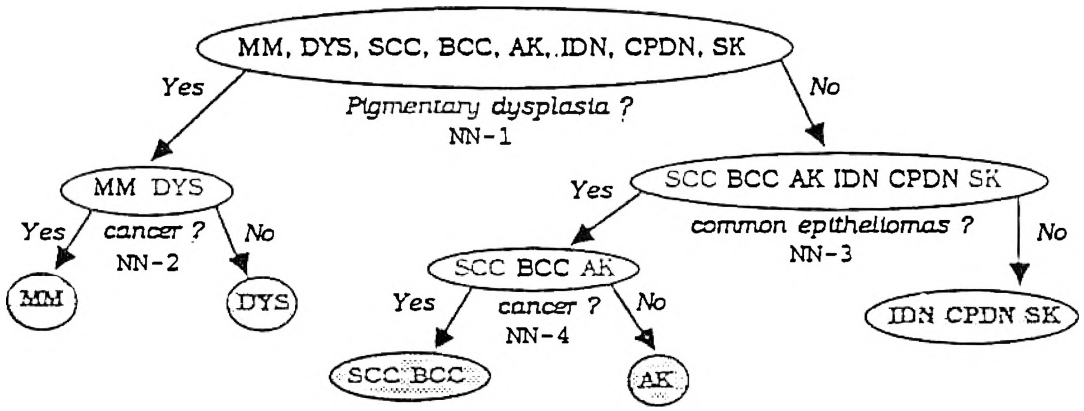

Figure 12 Hierarchical Diagnosis of Skin Tumors Using Four Neural Networks

APPENDIX A.

LISTING OF PROGRAM SOURCE

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <conio.h>
#include <ctype.h>
#include <string.h>
#include <iostream.h>

#define ESC   27
#define ITEMS  8

// object types
class BaseNet {          // A basic Neural Network type
                        // Includes matrix methods
public:
   float temp,
         Eta,           // default learning rate
         Alpha,         // default momentum factor
         ErrorLevel,    // acceptable error level
         Error;         // latest sum squared error value
   char  KeyboardRequest; // true when key pressed
   int   ErrorFreq,     // error reporting frequency
         nInputNodes,   // number of input nodes
         nHiddenNodes,  // number of hidden nodes
         nOutputNodes,  // number of output nodes
         nIterations,   // number of iterations
         nPatterns,     // number of patterns
         nRuns,         // number of runs (or input lines)
         H,             // index hidden layer
         I,             // index input layer
         J,             // index output layer
         P,             // index pattern number
         Q,             // index iteration number
         R;             // index run number
   FILE  *RunFile,        // RUN file
         *PatternFile,    // source pattern input file
         *WeightsInFile,  // initial weight file
         *WeightsOutFile, // final weight output file
         *ResultsFile,    // results output file
         *ErrorFile;      // error output file
   char  szResults[40];   // various filenames
   char  szError[40];
   char  szPattern[40];
   char  szWeights[40];
   char  szWeightsOut[40];

// Matrix
// typedefs and prototypes for dynamic storage of arrays
   typedef float *FLOATPTR;   // Pointer to a real
   typedef FLOATPTR VECTOR;   // A Vector: one column
   typedef FLOATPTR *MATRIX;  // A Matrix: two columns
//     typedef FLOATPTR MATRIX;  // A Matrix: two columns
// Network Layers
// Arrays for inputs, outputs, deltas, weights & target outputs
   MATRIX   Out0;         // input layer
   MATRIX   Out1;         // hidden layer
   MATRIX   Delta1;       // delta at hidden layer
   MATRIX   Delw1;        // change in weights input:hidden
   MATRIX   W1;           // weights input:hidden
   MATRIX   Out2;         // output layer
   MATRIX   Delta2;       // delta at output layer
   MATRIX   Delw2;        // change in weights hidden:output
   MATRIX   W2;           // weights hidden:output
   MATRIX   TargetOutput; // target output
   VECTOR   PatternID;    // identifier for each stored pattern

// Memory allocation methods
   void  AllocateVector(VECTOR *Vector, int nCols);
   void  AllocateColumns(FLOATPTR Matrix[], int nRows, int nCols);
   void  AllocateMatrix(MATRIX *pmatrix, int nRows, int nCols);
   void  FreeMatrix(MATRIX Matrix,  int nRows);

   BaseNet();          // constructor
  ~BaseNet() {};       // destructor
   virtual void Iterate(char Netname) {}; // abstract iteration loop
```

```
                                        // for any network
);

class BackProp : public BaseNet {  // Back Propagation network
public:
   BackProp()  {};
   ~BackProp() {};
   void Iterate(char Netname);  // iteration loop for this network
};

// BaseNet constructor initializes default fields.

BaseNet::BaseNet() {
    Eta    =  0.15,        // default learning rate
    Alpha =  0.075;        // default momentum factor
    ErrorFreq = 1;       // error reporting frequency
    ErrorLevel = 0.04;     // acceptable error level
    KeyboardRequest = 0;  // true when key pressed
);

// BaseNet methods

// Implementation of array allocation routines
// Allocate space for a vector of float cells,
// a one dimensional dynamic vector[cols]

void BaseNet::AllocateVector(VECTOR *Vector, int nCols)
{
   if ((*Vector = (VECTOR) calloc(nCols, sizeof(float))) == NULL)
   {
      cout << " Not enough memory!\n"; // If not, abort.
      exit(1);
   }
}

// Allocate space for a dynamic two dimensional matrix[rows][cols]
void BaseNet::AllocateColumns(FLOATPTR Matrix[], int nRows, int nCols)
{
   int  i;
   for (i = 0;  i < nRows;  i++)
      AllocateVector(&Matrix[i], nCols);
}

void BaseNet::AllocateMatrix(MATRIX *Pmatrix, int nRows, int nCols)
{
   if ((*Pmatrix = (MATRIX) calloc(nRows, sizeof(FLOATPTR))) == NULL)
   {
      cout << "Not enough memory!\n";
      exit(1);
   }
   AllocateColumns(*Pmatrix, nRows, nCols);
);

// Free the memory used by the Matrix
void BaseNet::FreeMatrix(MATRIX Matrix, int nRows)
{
   int   i;
   for (i = 0;  i < nRows;  i++)
      free(Matrix[i]);
   free(Matrix);
}

// Specific implementation of iteration loop
// for a back-propagation network

void BackProp::Iterate(char Netname) {
   for (R = 0;   R < nRuns;   R++)
   {
      // Read and parse the run specification line
      // to obtain information about this network.
      fscanf(RunFile,
         "%s %s %s %s %s %d %d %d %d %d %f %f",
         szResults,     // output results file
         szError,       // error output file
         szPattern,     // pattern input file
         szWeights,     // initial weights file
```

```
          szWeightsOut, // final weights output file
          &nPatterns,   // number of patterns to learn
          &nIterations, // number of iterations through the data
          &nInputNodes, // number of input nodes
          &nHiddenNodes,// number of hidden nodes
          &nOutputNodes,// number of output nodes
          &Eta,         // learning rate
          &Alpha);      // momentum factor

// Allocate dynamic storage for nodes and patterns.
      AllocateMatrix(&Out0,      nPatterns,    nInputNodes);
      AllocateMatrix(&Out1,      nPatterns,    nHiddenNodes);
      AllocateMatrix(&Out2,      nPatterns,    nOutputNodes);
      AllocateMatrix(&Delta2,    nPatterns,    nOutputNodes);
      AllocateMatrix(&Delw2,     nOutputNodes, nHiddenNodes + 1);
      AllocateMatrix(&W2,        nOutputNodes, nHiddenNodes + 1);
      AllocateMatrix(&Delta1,    nPatterns,    nHiddenNodes);
      AllocateMatrix(&Delw1,     nHiddenNodes, nInputNodes + 1);
      AllocateMatrix(&W1,        nHiddenNodes, nInputNodes + 1);
      AllocateMatrix(&TargetOutput,nPatterns,  nOutputNodes);
      AllocateVector(&PatternID, nPatterns);

      //ifstream WeightsInFile("WEIGHT.WTS");

//  Read the initial weight matrices.
      if ((WeightsInFile = fopen(szWeights,"r"))  ==  NULL)
      {
         cout << " Can't open file \n" <<  Netname << szWeights;
         exit(1);
      }

// read input:hidden weights
      for (H = 0;  H < nHiddenNodes;  H++)
         for (I = 0;  I <= nInputNodes;  I++)
         {
        //    WeightsInFile >> W1[H][I];
            fscanf(WeightsInFile, "%f", &W1[H][I]);
            printf("%f\n",W1[H][I]);
            Delw1[H][I] = 0.0;
         }

// read hidden:out weights
      for (J = 0;  J < nOutputNodes;  J++)
         for (H = 0;  H <= nHiddenNodes;  H++)
         {
            fscanf(WeightsInFile, "%f", &W2[J][H]);
            printf("%f\n",W2[J][H]);
            Delw2[J][H] = 0.0;
         }
      fclose(WeightsInFile);

// Read in all patterns to be learned.
      if ((PatternFile = fopen(szPattern, "r"))  ==  NULL)
      {
         cout << " Can't open file \n" << Netname << szPattern;
         exit(1);
      }

      for (P = 0;  P < nPatterns;  P++)
      {
         for (I = 0; I < nInputNodes; I++)
       //          if (fscanf(PatternFile,"%f", &Out0[P][I])!= 1)
              fscanf(PatternFile,"%f",&Out0[P][I]);
          //   goto AllPatternsRead;

// Read in targedt outputs for input patterns.
        for (J = 0; J < nOutputNodes; J++)
          { fscanf(PatternFile, "%f", &TargetOutput[P][J]);
           printf(" %f\n",TargetOutput[P][J]);}

// Read in identifier for each pattern.
        fscanf(PatternFile, "%f",  &PatternID[P]);
        printf("%f\n",PatternID[P]);
      }
      AllPatternsRead:      // Then, we're done.
      fclose(PatternFile);
```

```
      if (P < nPatterns)
      {
         cout << " Can't open file \n" <<  Netname << P << nPatterns;
         nPatterns = P;
      }

// Open error output file
      if ((ErrorFile = fopen(szError, "w"))  ==  NULL)
      {
         cout << " Can't open file \n" << Netname << szError;
         exit(1);
      }
       fprintf(stderr, nIterations > 1 ? "Training...\n" : "Testing\n");

//  Iteration loop
      for (Q = 0;  Q < nIterations;  Q++)
      {
         for (P = 0;  P < nPatterns;  P++)
         {

      // Hidden layer
      // Sum input to hidden layer over all
      // input-weight combinations
            for (H = 0;  H < nHiddenNodes;  H++)
            {
               float Sum = W1[H][nInputNodes];  // Begin with bias
               for (I = 0;  I < nInputNodes;  I++)
                  Sum   +=   W1[H][I]  *  Out0[P][I];

               // Compute output using sigmoid function.
               Out1[P][H] = 1.0 / (1.0 + exp(-Sum));
            }

       // Output layer
            for (J = 0;  J < nOutputNodes;  J++)
            {
               float  Sum = W2[J][nHiddenNodes];
               for (H = 0;  H < nHiddenNodes;  H++)
                  Sum  +=   W2[J][H]  *  Out1[P][H];
               // Compute output using sigmoid function.
               Out2[P][J]  =  1.0  /  (1.0  +  exp(-Sum));
            }

          // Delta output
          // Calculate deltas for each output unit for each pattern.
            for (J = 0;  J < nOutputNodes;  J++)
               Delta2[P][J] = (TargetOutput[P][J] - Out2[P][J])  *
                              Out2[P][J] * (1.0 - Out2[P][J]);

          // Delta hidden
            for (H = 0;  H < nHiddenNodes;  H++)
            {
               float  Sum = 0.0;
               for (J = 0;  J < nOutputNodes;  J++)
                  Sum += Delta2[P][J] * W2[J][H];
               // Compute output using sigmoid function.
               Delta1[P][H] = Sum * Out1[P][H] * (1.0 - Out1[P][H]);
            }
         }

          // Adapt weights hidden:output
          for (J = 0;  J < nOutputNodes;  J++)
         {
            float  Dw;        // delta weight
            float  Sum = 0.0;

            // Sum of deltas for each output node for one epoch
            for (P = 0;  P < nPatterns;  P++)
               Sum  +=  Delta2[P][J];

            // Calculate new bias weight for each output unit
            Dw   =   Eta * Sum  +  Alpha * Delw2[J][nHiddenNodes];
            W2[J][nHiddenNodes]   +=   Dw;
            Delw2[J][nHiddenNodes] =   Dw;  // delta for bias

            // Calculate new weights
```

```
            for (H = 0;  H < nHiddenNodes;  H++)
            {
               float  Sum = 0.0;
               for (P = 0;  P < nPatterns;  P++)
                  Sum  +=  Delta2[P][J] * Out1[P][H];
               Dw          =  Eta * Sum  +  Alpha * Delw2[J][H];
               W2[J][H]    += Dw;
               Delw2[J][H] =  Dw;
            }
         }

            // Adapt weights input:hidden
         for (H = 0;  H < nHiddenNodes;  H++)
         {
            float  Dw;                    // delta weight
            float  Sum = 0.0;
            for (P = 0;  P < nPatterns;  P++)
               Sum  +=  Delta1[P][H];

            // Calculate new bias weight for each hidden unit
            Dw   =   Eta * Sum  +  Alpha * Delw1[H][nInputNodes];
            W1[H][nInputNodes]   +=   Dw;
            Delw1[H][nInputNodes] =   Dw;

            // Calculate new weights
            for (I = 0;  I < nInputNodes;  I++)
            {
               float  Sum = 0.0;
               for (P = 0;  P < nPatterns;  P++)
                  Sum  +=  Delta1[P][H] * Out0[P][I];
               Dw          =   Eta * Sum  +  Alpha * Delw1[H][I];
               W1[H][I]    +=  Dw;
               Delw1[H][I] =   Dw;
            }
         }

 // Watch for keyboard requests
         if (kbhit())
         {
             int    c = getch();
             if ((c = toupper(c))  == 'E')
                KeyboardRequest++;
             else if (c == ESC)
                break;  // End if ESC request
         }

// Sum Squared Error
         if (KeyboardRequest || (Q % ErrorFreq == 0))
         {
            for (P = 0, Error = 0.0; P < nPatterns;   P++)
            {
               for (J = 0;  J < nOutputNodes;  J++)
               {
                  float  Temp   =   TargetOutput[P][J] - Out2[P][J];
                  Error += Temp * Temp;
               }
            }

            // Average error over all patterns
            Error  /=  nPatterns * nOutputNodes;

            // Print iteration number and  error value
            fprintf(stderr,"Iteration %5d/%-5d  Error %f\n",
               Q, nIterations, Error);
            KeyboardRequest = 0;

            if (Q % ErrorFreq == 0)
               fprintf(ErrorFile, "%d %f\n", Q, Error); // to file
            // Terminate when error satisfactory
            if (Error < ErrorLevel)
               break;
         }
      }
      // End iterate loop

      // Display error, iterations, etc.
```

```
        for (P = 0, Error = 0.0; P < nPatterns;  P++)
        {
           for (J = 0; J < nOutputNodes;  J++)
           {
             float Temp = TargetOutput[P][J] - Out2[P][J];
             Error += Temp * Temp;
           }
        }

        // Average error over all patterns
        Error  /=  nPatterns *nOutputNodes;

        // Print final iteration number and error value
        fprintf(stderr, "Iteration %5d/%-5d  Error %f\n", Q,
              nIterations, Error); /* to screen */
        fclose(ErrorFile);

        // Print final weights
        if ((WeightsOutFile = fopen(szWeightsOut,"w"))  ==  NULL)
        {
           cout << " Can't write file\n" << Netname << szWeightsOut;
           exit(1);
        }

        for (H = 0;  H < nHiddenNodes; H++)
           for (I = 0;  I <= nInputNodes; I++)
              fprintf(WeightsOutFile, "%g%c", W1[H][I],
                     I%ITEMS==ITEMS-1 ? '\n':' ');

        for (J = 0;  J < nOutputNodes;  J++)
           for (H = 0;  H <= nHiddenNodes;  H++)
              fprintf(WeightsOutFile,  "%g%c", W2[J][H],
                     J%ITEMS==ITEMS-1 ? '\n':' ');

        fclose(WeightsOutFile);

        // Print final activation values
        if ((ResultsFile = fopen(szResults,"w"))  ==  NULL)
        {
           cout << " Can't write file \n" << Netname << szResults;
           ResultsFile = stderr;
        }

        // Print final output vector
        for (P = 0;  P < nPatterns;  P++)
        {
  //       cout << ResultsFile << P;
           for (J = 0;  J < nOutputNodes;  J++)
              cout << Out2[P][J]<< endl;

           //  temp=Out2[P][J];
        //          printf("%f",temp);
         //          cout<<Out2[P][J];
      //   cout << ResultsFile << "\n" << PatternID[P];
        //        cout<<"\n"<< PatternID[P];
        }
        fclose(ResultsFile);

    // Free memory used for Matrix
        FreeMatrix(Out0, nPatterns);
        FreeMatrix(Out1, nPatterns);
        FreeMatrix(Delta1, nPatterns);
        FreeMatrix(Delw1, nHiddenNodes);
        FreeMatrix(W1, nHiddenNodes);
        FreeMatrix(Out2, nPatterns);
        FreeMatrix(Delta2, nPatterns);
        FreeMatrix(Delw2, nOutputNodes);
        FreeMatrix(W2, nOutputNodes);
        FreeMatrix(TargetOutput, nPatterns);
        free(PatternID);
     }
     fclose(RunFile);  // Close Run file
}

// Main program : creates and runs a BackProp Network
```

```
void main(int argc, char *argv[]) {

  BackProp Bp;              // Instance of a BackProp network
  char *Netname  =  *argv;  // netname is read from argument list

// Read arguments from DOS command line
for (; argc > 1;  argc--)
{
   char *arg = *++argv;
   if (*arg  !=  '-')
     break;
   switch (*++arg)
   {
    case 'e':   sscanf(++arg,  "%d",  &Bp.ErrorFreq);   break;
    case 'd':   sscanf(++arg,  "%f",  &Bp.ErrorLevel);  break;
    default:    break;
   }
}
if (argc < 2)
{
   fprintf(stderr, "Usage:  %s (-en -df) runfilename\n",  Netname);
   fprintf(stderr, "   -en   =>  report error every n iterations\n");
   fprintf(stderr, "   -df   =>  done if sum squared error < f\n");
   exit(1);
}

// Open run file for reading
if ((Bp.RunFile = fopen(*argv, "r"))  ==  NULL)
 {
   cout << " Can't open file \n" << Netname << *argv;
   exit(1);
}


   fscanf(Bp.RunFile, "%d", &Bp.nRuns); // Scan for no. of runs
   Bp.Iterate(*Netname);                // Iterate a BackProp network.
);
```

APPENDIX B.

IMAGE SET WITH FEATURES

```
! Image Numbers and Features
! Feature Number and Feature
! 1 - Irregularity  Index
! 2 - Percent Asymmetry
! 3,4,5 - Variance of Red Green Blue Planes
! 6,7,8 - Relative chromaticity
! 9,10,11 - Spherical Transform -Length, Angle A, Angle B
! 12,13,14 -Lightness, Chroma, Hue
! 15 - Elevation greater than 1mm
! 16 - Area greater than 6mm


!1062n.pic -sk1
1.031 5.733 19 10 11 0.0691178 -0.0328287 -0.0362891 164 70 19 58 42 15
& 1 1 0


!1069n.pic -sk 2
1.017 8.884 17 10 8 0.00135021 -0.00397274 0.00262253 164 75 16 56 50 20
& 1 1 0

!1070n.pic -sk 3 ()
1.128 10.451 13 5 6 -0.0307848 0.00723781 0.023547 144 66 23 57 34 17
& 1 1 0

!1080n.pic -sk 4 ()
1.193 19.798 17 9 7 -0.0389408 0.000944793 0.037996 150 64 25 58 30 36
& 1 1 0

!1083n.pic  -sk 5 ( )
1.206 18.050 14 8 6 -0.0223464 -0.0023347 0.0246811 165 65 26 62 29 12
& 1 1 0

!1087n.pic -sk 6  ()
1.034 8.321 9 7 6 0.0440858 -0.0213571 -0.0227287 213 73 22 68 41 32
& 0 1 0

!1090n.pic -sk 7  ()
1.02 1 8.321 11 12 9 0.116759 -0.0603349 -0.0564245 241 74 21 71 44 31
& 1  1 0

!1091n.pic -sk 8
1.102 7.321 13 7 7 0.0548701 -0.0408009 -0.0140693 170 70 21 61 38 19
& 1 1 0

!1092ncmp.pic -sk 9
1.056 12.174 13 6 7 0.013108 -0.0147382 0.00163018 156 74 15 54 50 14
& 0 1 0


!1094ncmp.pic -sk 10
1.027 4.9049 6 7 7 0.0325351 -0.0129994 -0.0195357 223 67 24 59 39 32
&  0 1 0

!2007ncmp.pic -sk 11
1.012 4.125 7 4 3 0.0960578 -0.0341857 -0.0618721 168 80 13 54 61 22
& 1 1 0

!2037ncmp.pic -sk 12
```

```
1.352 5.886 18 7 5 0.346447 -0.1944 -0.152047 85 88 3 32 90 41
& 1 1 0

!2059n.pic -sk 13
1.136 13.475 14 3 3 0.191546 -0.112665 -0.0788812 38 86 13 20 62 34
& 1 1 0

!2087ncmp.pic -sk 14
1.157 8.455 13 6 7 -0.00686729 -0.00569877 0.0125661 156 72 15 54 50 14
& 1 0 0

!2105ncmp.pic  -sk 15
1.041 9.632 12 14 8 -0.115576 0.0538567 0.0617192 64 21 61 23 12 12
& 1 0 0

!211ncmp.pic -sk 16
1.015 4.432 1 14 12 0.0516641 -0.0246662 -0.0269979 319 70 34 88 32 73
& 1 1 0

!2160ncmp.pic -sk 17
1.168 6.930 17 9 7 -0.27037 0.134001 0.136369 223 67 24 59 39 32
& 1 1 0

!2161ncmp.pic -sk 18
1.048 2.472 1 14 12 -0.29676 0.176275 0.120485 64 21 61 23 12 12
& 1 1 0

!1286n.pic -sk 19
1.120 9.265 4 3 3 0.101622 -0.0475972 -0.0540251 176 78 13 56 59 21
& 0 1 0


!2189ncmp.pic  -sk 20
1.355 11.250 8 4 4 -0.0234213 0.00143669 0.0219846 103 75 13 44 49 12
&  1 1 0

!2190ncmp.pic  -sk 21
1.101 8.213 7 1 8 0.0616013 -0.0360363 -0.0255651 112 76 11 41 42 11
& 1 1 0


!1056n.pic sk -22
1.153 11.905 11 2 3 -0.0131334 0.00503908 0.00809427 120 69 20 51 37 9
& 1 1 0

!1057n.pic sk -23
1.079 12.217 20 7 9 -0.0354573 0.0147691 0.0206882 175 66 22 61 38 22
&  1 0 0

!1061n.pic sk -24
1.032 8.321 5 7 8 0.0535 -0.0216394 -0.0318606 256 68 22 72 41 13
& 1  0 0

!1066n.pic sk -25
1.021 10.543 22 12 8 0.0623497 -0.0453598 -0.0169899 179 74 21 62 41 31
& 1 0 0


!1084n.pic sk -27
1.024 8.321 16 8 7 0.0640028 -0.0433885 -0.0206143 168 71 21 60 38 21
& 1 0 0
```

```
!1070n sk -33
1.128 10.541 30 12 8 0.0843952 -0.0587258 -0.0256694 21 24 38 31 69 38
& 1 0 0

!1071  sk -34
1.036 3.194 22 12 7 0.10456 -0.0310324 -0.0735272 17 15 39 32 57 37
& 1 0 0

!1072 sk -35
1.073 6.675 26 14 5 0.168369 -0.115165 -0.0532037 20 18 37 31 60 33
& 1 0 0

!1073 sk -36
1.052 6.482 21 15 6 0.155799 -0.123364 -0.032435 31 19 33 30 69 35
& 1 0 0

!1074 sk -37
1.036 4.800 33 24 9 0.220329 -0.163921 -0.0564075 22 31 31 32 72 39
& 1 0 0

!1075 sk -38
1.253 12.523 23 14 7 0.157201 -0.121593 -0.0356081 21 34 32 31 79 41
& 1 0 0

!1076 sk -39
1.141 7.038 32 15 7 0.199476 -0.207951 0.00847509 21 34 32 32 77 37
& 1 0 0

!1285 sk -40
1.042 9.900 29 12 4 0.230517 -0.261197 0.03068 21 31 41 31 72 34
& 1 1 0

!2185ncmp.pic  -sk 41
1.103 5.023 6 2 3 0.1283 -0.0684647 -0.059835 157 82 9 50 69 24
& 1 1 0

!1100n.pic sk -42
1.073 9.873 20 10 12 0.0749288 -0.0321937 -0.042735 185 74 15 59 53 15
& 1 0 0

!1097ncmp.pic -idn 1
1.160 20.732 15 3 3 -0.0133534 0.000698402 0.012655 153 72 17 55 46 14
& 0 0 0

!1103n.pic -idn 2
1.174 8.539 6 3 3 0.0661081 -0.0302028 -0.0359054 189 75 14 58 57 15
& 0 1 0

!1104n.pic  -idn 3
1.146  10.609 6 3 4 0.0872334 -0.0372876 -0.0499458 234 76 13 63 64 17
& 0 1 0

!1107ncmp.pic -idn 4
1.033 4.491 9 7 4 0.263736 -0.131639 -0.132097 133 80 15 51 52 32
& 1 1 0

!1112n.pic -idn 5
1.017 2.501 24 14 9 0.013667 -0.0223494 0.0086824 146 76 20 56 41 32
& 1 0 0
```

```
!1115n.pic -idn 6
1.101 8.016 23 11 9 0.0527529 -0.0609775 0.00822463 145 72 21 54 43 31
& 1 0 0

!1119n.pic  -idn 7
1.017 5.699 7 8 8 0.0630696 -0.0340714 -0.0289982 213 76 16 63 55 30
& 1 0 0

!115ncmp.pic -idn 8
1.049 7.317 13 7 4 0.183424 -0.0934123 -0.0900115 139 83 11 49 64 33
& 1 0 0

!116n.pic  -idn 9
1.096 7.123 8 8 7 0.0821575 -0.0497772 -0.0323803 221 74 19 67 48 25
& 0 1 0

!1161n.pic -idn 10
1.035 6.703 8 15 13 0.0307125 -0.0208465 -0.009866 257 71 26 76 35 38
& 0 0 0

!1260n.pic  -idn 11
1.057 11.404 10 8 8 0.0519384 -0.036266 -0.0156724 178 72 19 60 44 18
& 1 1 0

!1274n.pic -idn 12
1.083 8.058 5 1 1 -0.0677258 0.0186354 0.0490905 136 67 22 55 34 10
& 0 1 0

!1277n.pic -idn 13
1.178 13.033 9 2 2 -0.033558 0.00336927 0.0301887 93 72 19 45 35 17
& 0 1 0

!1278n.pic -idn 14
1.114 3.756 9 3 4 -0.0300774 0.00930613 0.0207713 120 69 19 51 38 13
& 1 0 0

!1280n.pic -idn 15
1.120 5.499 9 4 5 -0.031015 0.00125313 0.0297619 119 76 14 48 49 16
& 0 1 0


!241ncmp.pic  -idn 16
1.081 2.828 26 15 7 0.18071 -0.0825137 -0.0981966 155 84 18 57 59 54
&  0 1 0


!2019 2 idn -17
1.041 9.645 7 7 6 0.116756 -0.0631656 -0.0535906 138 83 12 49 63 34
& 0 0 0

!2020 3 idn -18
1.008 5.963 9 7 5 0.146491 -0.0794678 -0.0670234 133 82 11 48 62 30
& 0 0 0

!2021 4 idn -19
1.046 3.140 11 6 4 0.113997 -0.0585859 -0.0554113 107 87 7 40 82 40
& 0  0 0

!2052 6595 5 idn -20
1.114 9.132 11 5 4 0.108052 -0.0614302 -0.0466216 131 84 8 45 71 28
& 0 0 0
```

```
!2068 22185 6 idn -21
1.065 8.227 10 6 5 0.0716841 -0.0437076 -0.0279765 186 82 9 54 72 23
& 0  0 0

!2071 16501 7 idn -22
1.019 9.639 8 4 4 0.0932401 -0.0441357 -0.0491044 130 84 7 44 72 28
& 0    0 0

!2144 49941 idn -23
1.187 12.122 10 6 6 0.0821358 -0.0388926 -0.0432432 134 77 12 49 56 16
& 0  0 0

!27 29214 idn -24
1.040 5.590 15 9 6 0.131055 -0.078216 -0.0528394 154 83 9 50 69 29
& 0  0 0

!1256n.pic idn -25
1.145 5.08 21 6 5 0.120136 -0.0680946 -0.0520415 167 76 15 56 53 18
& 1 1 0

!1257n.pic idn -26
1.053 4.05 20 8 8 0.0755894 -0.0377947 -0.0377947 174 75 15 57 53 16
&1  0 0

!1259n.pic idn -27
1.114 5.468 17 10 10 0.0939308 -0.0469654 -0.0469654 187 76 14 58 57 17
& 1 0 0

!1262n.pic idn -28
1.075 8.315 6 7 7 0.146951 -0.0772348 -0.069716 189 76 13 58 59 15
& 1 0 0

!1263n.pic idn -29
1.098 12.453 23 9 7 0.105208 -0.0554246 -0.0497834 147 75 17 54 45 22
& 1  0 0


!1264n.pic idn -30
1.088 6.925 9 7 7 0.0446026 -0.0235533 -0.0210492 187 75 15 59 55 18
& 1 0 0

!1266n.pic idn -31
1.055 7.989 9 10 11 0.105228 -0.0519599 -0.0532679 210 74 16 63 54 18
& 1  0 0

!1267n.pic idn -32
1.143 11.999 9 5 6 0.0905609 -0.0455797 -0.0449812 215 76 13 61 61 16
& 1 0 0

!1268n.pic idn -33
1.045 4.818 21 7 6 0.110261 -0.0593543 -0.0509067 162 74 17 57 47 18
& 1 0 0

!1272n.pic idn -34
1.011 4.432 11 7 8 0.0464534 -0.0302478 -0.0162057 195 74 16 61 53 17
& 1 0 0

!1275n.pic idn -35
1.135 9.982 18 8 9 0.0890631 -0.0519898 -0.0370733 163 75 14 55 53 15
& 1 0 0
```

```
!2022ncmp.pic idn -36
1.032 5.321 13 7 6 0.0503824 -0.0275247 -0.0228576 132 83 12 49 63 37
& 1 0 0

!207ncmp.pic idn -37
1.065 5.321 20 18 13 0.241703 -0.0891783 -0.152524 216 78 20 67 51 43
& 1 0 0


!1101n.pic -38 idn
1.021 6.211 17 12 12 0.0871868 -0.0509664 -0.0362205 222 72 16 64 53 13
& 1 0 0

!1108n  -39 idn
1.032 6.432 17 9 10 0.115745 -0.0630272 -0.0527179 202 74 14 60 57 14
& 1 0 0

!1113n  -40 idn
1.042 7.322 8 14 12 0.0986735 -0.0090261 -0.0950474 238 70 23 70 40 28
& 1  0 0


!2016  idn -  41
1.217 6.741 10 11 10 0.163615 -0.0854795 -0.0781355 159 79 11 52 63 21
& 0 0 0


!133ncmp.pic -dys 1
1.062 8.873 5 21 17 0.145339 -0.0506433 -0.0946962 269 75 26 77 43 53
& 0 1 0

!141ncmp.pic -dys2
1.105 5.658 22 14 7 0.170337 -0.0828349 -0.0875022 136 84 13 50 61 42
& 1 1 0

!143ncmp.pic -dys 3
1.157 2.784 26 12 7 0.214605 -0.106758 -0.107848 137 84 12 49 63 40
& 0 1 0

!156ncmp.pic -dys 4
1.043 4.766 10 14 9 0.160594 -0.0688262 -0.0917682 231 80 21 70 55 52
& 0 1 0

!157ncmp.pic -dys 5
1.120 8.718 23 12 6 0.174069 -0.0826856 -0.091383 150 83 17 55 56 51
& 0 1 0

!158ncmp.pic  -dys 6
1.132 16.294 18 11 8 0.204943 -0.105352 -0.0995905 180 84 10 54 73 35
& 0 1 0

!159ncmp.pic -dys 7
1.234 17.649 18 27 19 0.140179 -0.0449167 -0.0952622 234 78 25 73 49 61
& 0 1 0

!160ncmp.pic -dys 8
1.227 19.758 18 5 1 0.277632 -0.137794 -0.139839 116 89 5 40 98 48
& 0 1 0


!218ncmp.pic  -dys 9
```

```
1.140 17.866 17 28 25 0.150561 -0.0567199 -0.0938408 238 72 27 74 35 48
& 0 1 0


!219ncmp.pic  -dys 10
1.098 4.250 9 5 3 0.132369 -0.0712994 -0.0610694 162 84 15 56 63 45
& 0 1 0


!220ncmp.pic  -dys 11
1.184 1.852 17 9 5 0.109123 -0.0722351 -0.0368875 166 82 15 56 59 38
& 0 1 0

!475ncmp.pic  -dys 12
1.272 25.290 29 22 12 0.13804 -0.0698198 -0.0682204 206 81 21 66 55 56
& 0 0 0

!482ncmp.pic  -dys 13
1.071 9.600 27 12 5 0.206347 -0.119573 -0.0867738 101 86 11 42 71 51
& 0 0 0

!484ncmp.pic -dys 14
1.031 8.342 25 13 7 0.0539652 -0.0339897 -0.0199755 155 77 18 56 46 31
& 0  0    0

!486ncmp.pic -dys 15
1.040 7.947 13 7 4 0.125833 -0.0723079 -0.053525 146 79 17 55 49 35
& 0 0 0

!487ncmp.pic -dys 16
1.053 8.480 13 11 7 0.0387884 -0.0209809 -0.0178075 129 79 20 54 45 49
& 0 0 0

!49ncmp.pic  -dys 17
1.102 11.510 20 20 16 0.0776418 -0.0506528 -0.026989 109 70 25 51 27 30
& 0 1 0

!490ncmp.pic -dys 18
1.119 14.532 3 0 0 0.226667 -0.173333 -0.0533333 23 90 21 13 72 44
& 0 1 0

!496ncmp.pic -dys 19
1.047 9.338 6 10 5 0.273172 -0.155574 -0.117598 83 85 16 42 64 59
& 0 0 0

!497ncmp.pic-dys 20
1.032 7.431 7 2 0 0.279339 -0.204959 -0.0743802 54 89 0 23 92 45
& 0 0 0

!499ncmp.pic-dys 21
1.008 5.586 18 24 18 -0.00818678 0.00596321 0.00222357 135 77 24 56 41 59
& 0 0 0

!500ncmp.pic-dys 22
1.098 13.380 10 15 10 0.158599 -0.0872711 -0.071328 159 78 18 57 49 37
& 0 0 0


!216ncmp.pic -dys 23
1.086 12.711 10 10 6 0.126032 -0.0737281 -0.0523043 216 81 15 63 63 37
& 0 1 0
```

!502ncmp.pic-dys 24
1.012 9.872 14 4 1 0.287735 -0.185373 -0.102362 56 89 2 25 87 47
& 0 1 0

!503ncmp.pic-dys 25
1.012 8.765 10 5 2 0.141607 -0.0763629 -0.0652446 171 79 25 64 45 62
& 0 0 0

!504ncmp.pic-dys 26
1.032 8.3421 14 8 3 0.117862 -0.0829299 -0.0349323 107 79 16 47 46 36
& 0 0 0

!506ncmp.pic-dys 27
1.032 7.321 24 13 7 0.188732 -0.100218 -0.0885141 135 86 13 50 69 50
& 0 0 0

!507ncmp.pic-dys 28
1.021 6.443 25 16 10 0.141358 -0.0672615 -0.0740963 100 83 18 46 54 58
& 0 0 0

!508ncmp.pic-dys 29
1.045 8.222 16 24 21 0.127917 -0.0341737 -0.0937431 166 72 24 61 34 34
& 0 0 0

!509ncmp.pic -dys 30
1.101 5.321 21 14 10 0.140047 -0.0712651 -0.0687816 114 77 20 51 38 40
& 0 0 0

!510ncmp.pic-dys 31
1.005 9.082 22 13 8 0.133109 -0.0710516 -0.0620577 120 80 16 49 49 37
& 0 0 0

!511ncmp.pic -dys 32
1.0998 6.504 15 7 3 0.250911 -0.148865 -0.102047 72 88 5 31 83 47
& 0 0 0

!513ncmp.pic-dys 33
1.018 7.247 21 16 9 0.193961 -0.10456 -0.0894005 105 85 14 44 68 54
& 0 0 0

!514ncMp.pic -dys 34
1.044 10.826 22 15 7 0.194506 -0.109888 -0.0846171 96 85 14 43 66 54
& 0 0 0

!515ncmp.pic-dys 35
1.014 4.368 7 12 6 0.205376 -0.106605 -0.0987711 105 85 18 48 66 63
& 0 0 0

!516ncmp.pic -dys 36
1.095 3.662 15 8 5 0.195002 -0.107989 -0.0870132 69 81 16 38 44 43
& 0 0 0

!517ncmp.pic-dys 37
1.118 15.775 21 10 4 0.201378 -0.127981 -0.0733965 73 88 6 32 81 49
& 0 0 0

!518ncmp.pic -dys 38
1.068 5.029 11 8 5 0.157976 -0.0847907 -0.0731849 136 86 12 50 71 47
& 0 0 0

```
!521ncmp.pic-dys 39
1.096 10.993 11 10 6 0.216075 -0.123325 -0.0927504 146 84 15 54 63 48
& 0 0 0

!522ncmp.pic -dys 40
1.163 11.136 22 19 9 0.205653 -0.109019 -0.0966339 97 85 18 45 62 64
& 0 0 0

!523ncmp.pic -dys 41
1.118 14.871 8 16 9 0.121843 -0.0789141 -0.0429293 120 80 24 54 46 65
& 0 0 0

!524ncmp.pic -dys 42
1.066 9.132 5 10 7 0.0829462 -0.0529668 -0.0299794 118 84 16 49 64 52
& 0 0 0

!526ncmp.pic-dys 43
1.010 4.314 14 13 10 0.119831 -0.0591585 -0.0606725 146 81 16 54 54 39
& 0 0 0

!527ncmp.pic -dys 44
1.090 6.920 20 13 6 0.200477 -0.109688 -0.090789 96 86 14 43 68 57
& 0 0 0

!528ncmp.pic -dys 45
1.021 7.642 13 12 7 0.0994053 -0.0525721 -0.0468332 130 83 17 52 59 53
& 0 0 0

!529ncmp.pic-dys 46
1.208 17.275 16 8 4 0.138636 -0.0740909 -0.0645455 88 87 13 41 70 54
& 0  0 0

!530ncmp.pic-dys 47
1.302 11.303 4 7 5 0.213229 -0.132196 -0.0810335 120 82 14 48 55 38
&0 0 0

!531ncmp.pic-dys 48
1.160 7.600 13 6 2 0.322011 -0.210499 -0.111511 58 89 7 30 78 54
&0 0  0

!532ncmp.pic -dys 49
1.246 14.695 18 10 7 0.205027 -0.105606 -0.0994211 107 85 11 43 66 44
& 0 1 0

!533ncmp.pic-dys 50
1.031 11.285 17 10 4 0.051419 -0.0373818 -0.0140372 80 79 17 41 41 37
& 0  1 0

!534ncmp.pic-dys 51
1.155 13.234 28 21 11 0.134924 -0.0692665 -0.0656577 128 83 23 54 54 67
& 0 0 0

!535ncmp.pic -dys 52
1.015 12.214 12 12 6 0.0594883 -0.0368852 -0.0226031 96 84 18 46 59 61
& 0 0 0

!537ncmp.pic -dys 53
1.031 5.177 22 13 6 0.257882 -0.131213 -0.126669 100 87 12 43 72 54
& 1 1 0

!538ncmp.pic-dys 54
```

1.168 26.471 15 5 2 0.320127 -0.189237 -0.13089 98 89 3 35 100 47
& 0 1 0

!539ncmp.pic -dys 55
1.015 7.508 17 8 5 0.101143 -0.0586939 -0.042449 96 84 12 42 57 39
& 0 0 1

!540ncmp.pic -dys 56
1.020 10.072 18 15 9 0.142786 -0.078607 -0.0641791 98 84 13 42 65 47
& 0 1 0

!541ncmp.pic-dys 57
1.066 12.906 18 8 5 0.197635 -0.102055 -0.09558 92 87 10 39 75 48
& 0 0 0

!542ncmp.pic-dys 58
1.145 18.876 21 14 7 0.18842 -0.100308 -0.0881125 93 86 15 43 67 58
& 1 0 0

!543ncmp.pic-dys 59
1.114 14.183 11 11 7 0.123471 -0.0629612 -0.0605102 169 79 22 62 46 50
& 0 1 0

!544ncmp.pic-dys 60
1.113 13.279 17 12 7 0.208518 -0.147572 -0.0609461 108 85 14 46 63 49
& 0 1 0

!545n.pic-dys 61
1.288 11.643 22 28 19 0.0809524 -0.0449735 -0.0359788 192 70 35 70 26 76
& 0 1 0

!546n.pic -dys 62
1.038 4.373 17 6 3 0.223604 -0.130697 -0.0929074 104 86 10 42 69 44
& 0 1 0

!547n.pic -dys 63
1.085 5.445 21 10 7 0.18869 -0.0928571 -0.0958333 83 81 12 38 49 28
& 0 1 0

!548n.pic-dys 64
1.240 18.830 17 10 8 0.179803 -0.0909438 -0.0888588 146 77 14 53 52 24
& 0 1 0

!549n.pic -dys 65
1.059 4.686 26 14 10 0.0660927 -0.0307781 -0.0353146 129 71 23 54 31 25
& 0 1 0

!550n.pic-dys 66
1.066 11.685 18 9 4 0.250476 -0.134286 -0.11619 110 86 13 45 66 49
& 0 1 0

!551ncmp.pic-dys 67
1.063 6.800 14 13 10 0.18094 -0.12279 -0.0581502 133 80 18 53 49 44
& 0 1 0

!552ncmp.pic -dys 68
1.016 6.320 21 20 13 0.155285 -0.0897358 -0.0655488 114 82 20 50 52 56
& 0 1 0

!553ncmp.pic-dys 69
1.180 11.580 23 15 8 0.170833 -0.0902778 -0.0805556 106 85 17 47 64 60

```
& 0  1 0

!554ncmp.pic -dys 70
1.061 12.236 23 11 5 0.124734 -0.0615991 -0.0631353 101 83 17 46 50 50
& 0  1 0

!555ncmp.pic-dys 71
1.423 11.672 15 6 3 0.216049 -0.123457 -0.0925926 71 88 5 31 80 43
& 0  1 0

!556ncmp.pic-dys 72
1.043 11.554 6 1 2 0.0481713 -0.0448508 -0.00332045 81 74 17 42 38 19
& 0  1 0

!558ncmp.pic -dys 73
1.313 25.546 16 12 4 0.288543 -0.174943 -0.113599 65 88 9 32 77 56
& 0 1 0

!559ncmp.pic -dys 74
1.051 8.623 11 4 0 0.198925 -0.150538 -0.0483871 66 89 4 29 90 51
& 0  1 0

!561ncmp.pic -dys 75
1.089 8.243 10 12 8 0.153428 -0.0718737 -0.0815538 103 81 19 48 47 49
& 0  1 0

!562ncmp.pic -dys 76
1.089 5.180 7 13 8 0.119906 -0.0702758 -0.0496306 142 80 23 58 46 57
& 0  1 0

!498ncmp.pic dys nevi - 77
1.093 9.018 14 3 1 0.288557 -0.168339 -0.120219 64 89 2 27 90 45
& 0  1 0

!495ncmp.pic dys  nevi -78
1.066 18.638 8 8 3 0.249353 -0.165277 -0.0840759 61 88 8 31 75 51
& 0  1 0

!494ncmp.pic dys nevi - 79
1.003 3.651 9 12 7 0.2125 -0.116429 -0.0960714 82 83 17 42 56 55
& 0  1 0

!493ncmp.pic dys. nevi - 80
1.053 6.838 10 10 6 0.25 -0.134615 -0.115385 60 86 14 34 62 57
& 0 1 0

!489ncmp.pic dys. nevi - 81
1.031 5.388 10 18 10 0.128388 -0.0688155 -0.0595722 159 82 25 62 57 70
& 0  1 0

!488ncmp.pic dys. nevi - 82
1.021 6.580 23 19 13 0.10323 -0.0522811 -0.0509491 161 80 21 59 52 55
& 0  1 0

!44ncmp.pic dys. nevi - 83
1.298 11.543 10 19 19 0.142043 -0.0577351 -0.0843075 116 67 31 55 19 45
& 0  1 0

!40 48612 dys. nevi -84
1.214 19.065 17 17 17 0.183959 -0.0714789 -0.11248 133 68 25 56 28 29
& 0 1 0
```

!41 12468 dys. nevi -85
1.157 16.32 23 24 22 0.163334 -0.074249 -0.0890849 176 70 25 63 31 31
& 0 1 0

!474 2147 dys. nevi -86
1.013 6.624 30 20 10 0.132258 -0.0707044 -0.0615532 127 81 20 53 47 53
& 0 1 0

!476 2528 dys. nevi -87
1.038 7.879 15 18 14 0.10009 -0.049968 -0.0501221 222 76 19 66 50 31
& 0 1 0

!478 2608 dys. nevi -88
1.146 6.119 10 18 18 0.127568 -0.0612033 -0.0663644 200 67 25 67 31 16
& 0 1 0

!479 2267 dys. nevi -89
1.139 12.896 33 20 9 0.170581 -0.0844125 -0.0861682 142 84 17 54 60 55
& 0 1 0

!480 6012 dys. nevi -90
1.143 7.384 23 22 13 0.180189 -0.101745 -0.0784441 181 80 19 62 52 49
& 0 1 0

!481 15840 dys. nevi -91
1.082 5.884 20 8 4 0.181986 -0.101299 -0.0806871 84 86 11 39 64 45
& 0 1 0

!483 90 dys. nevi -92
1.106 14.673 27 17 8 0.0707658 -0.0407852 -0.0299806 185 77 21 63 45 42
& 0 1 0

!484 1970 dys. nevi -93
1.008 3.700 23 12 7 0.056357 -0.0390921 -0.0172649 153 77 18 56 46 30
& 0 1 0

!485 1892 dys. nevi -94
1.066 12.019 21 14 8 0.0916667 -0.0495098 -0.0421569 146 81 17 55 52 45
& 0 1 0

!490 4509 dys. nevi -95
1.119 14.572 8 1 0 0.226667 -0.173333 -0.0533333 36 89 0 17 81 45
& 0 1 0

!500 1424 dys. nevi -96
1.098 13.380 10 15 10 0.158599 -0.0872711 -0.071328 159 78 18 57 49 37
& 0 1 0

!501ncmp.pic-dys 97
1.032 12.22 24 24 14 0.108512 -0.0597078 -0.0488047 146 76 28 60 34 63
& 0 0 0

!43ncmp.pic -dys 98
1.047 3.732 6 3 2 0.14157 -0.06196 -0.0796101 95 73 23 49 29 36
& 0 1 0

```
!303ncmp.pic -mel 1
1.002 10.314 41 24 17 0.021307 -0.0221893 0.000882383 149 75 22 57 37 37
& 0 1 1

!304ncmp.pic -mel 2
1.368 7.887 27 13 5 0.128994 -0.0631023 -0.0658918 104 85 17 47 57 56
& 1 1 1

!305ncmp.pic -mel 3
1.161 6.686 11 15 10 0.167401 -0.0964374 -0.0709634 208 83 10 57 77 32
& 0 1 1

!308ncmp.pic -mel 4
1.354 22.706 19 9 7 0.0622807 -0.0392756 -0.0230051 161 78 16 56 51 28
& 0 1 1

!315ncmp.pic  -mel 5
1.252 16.447 19 11 9 0.13216 -0.0693582 -0.0628019 109 82 12 44 56 32
& 0 1 1

!318ncmp.pic -mel 6
1.249 18.419 10 4 3 0.115542 -0.066803 -0.0487389 91 86 7 38 67 34
& 0 1 1

!320ncmp.pic -mel 7
1.203 21.469 20 12 7 0.133948 -0.0768223 -0.0571257 65 87 9 32 69 50
& 1 1  1

!324ncmp.pic -mel 8
1.592 10.968 14 7 4 0.222871 -0.106973 -0.115897 78 85 12 38 58 44
& 0 1 1

!325ncmp.pic -mel 9
1.225 19.299 18 6 5 0.140449 -0.0702247 -0.0702247 129 80 14 50 51 30
& 0 1 1

!330ncmp.pic -mel 10
1.137 11.828 14 8 5 0.189286 -0.0988095 -0.0904762 95 85 11 41 64 42
& 1 1 1

!333ncmp.pic  -mel 11
1.205 13.853 11 3 2 0.186992 -0.097561 -0.0894309 91 80 13 42 48 27
& 0 1 1

!334ncmp.pic   -mel 12
1.317 23.449 15 11 9 0.10816 -0.0528886 -0.055271 75 72 21 42 30 27
& 0 1 1

!335ncmp.pic -mel 13
1.131 26.770 25 8 6 0.0821502 -0.0487085 -0.0334417 121 83 12 47 58 35
&  0 1 1

!340n.pic   -mel 14
2.342 43.468 11 4 3 0.234465 -0.128411 -0.106054 71 85 11 36 54 39
& 0 1 1

!342ncmp.pic  -mel 15
1.727 18.187 31 25 14 0.110295 -0.0599168 -0.050378 164 79 23 61 45 54
& 0 1 1

!343ncmp.pic -mel 16
```

```
1.328 18.678 9 9 5 0.140825 -0.0858287 -0.0549962 215 84 15 63 70 46
& 0 1 1

!345ncmp.pic  -mel 17
1.647 29.141 11 26 21 0.0911901 -0.0402883 -0.0509018 267 72 28 78 36 52
& 0 1 1

!350ncmp.pic  -mel 18
1.157 11.195 14 7 4 0.314298 -0.157367 -0.156931 54 88 5 27 73 47
& 1 1 1

!352ncmp.pic -mel 19
1.809 38.39 26 18 11 0.148915 -0.0599619 -0.0889532 107 83 20 49 50 60
& 0 1 1

!353ncmp.pic  -mel 20
1.128 14.27 0 10 5 0.150327 -0.0782794 -0.0720474 92 79 16 44 43 34
& 0 1 1

!355ncmp.pic  -mel 21
1.315 12.05 22 14 9 -0.0017011 -0.00908638 0.0107875 162 80 23 61 48 59
& 0 0 1

!356ncmp.pic  -mel 22
1.538 13.44 26 21 15 0.126084 -0.0819356 -0.0441488 167 77 20 59 45 40
& 1 1 1

!357ncmp.pic  -mel 23
1.391 20.958 14 22 14 0.0967153 -0.0499176 -0.0467977 247 77 24 73 47 52
& 0 0 1

!358n.pic -mel 24
1.375 20.516 29 16 10 0.0399772 -0.0291483 -0.0108289 191 81 17 61 56 42
& 0 0 1

!359ncmp.pic  -mel 25
1.276 14.424 23 13 6 0.115062 -0.0531687 -0.061893 58 87 13 32 61 59
& 0 1 1

!361ncmp.pic  -mel 26
1.172 13.544 25 18 13 0.0601316 -0.0516219 -0.00850976 155 80 17 56 52 38
& 1 1 1

!362ncmp.pic  -mel 27
1.772 17.948 29 10 8 0.10652 -0.0461802 -0.0603403 102 82 15 45 49 40
& 1 1 1

!363ncmp.pic  -mel 28
1.194 9.613 9 16 12 0.0644723 -0.0281645 -0.0363078 270 74 30 80 38 66
& 0 1 1

!364n.pic    -mel 29
1.086 9.513 34 16 8 0.144444 -0.082846 -0.0615984 95 82 21 46 45 59
& 0 1 1

!366ncmp.pic    -mel 30
1.277 14.503 19 7 4 0.194758 -0.113564 -0.0811934 114 85 11 44 67 41
& 0 1 1

!368ncmp.pic   -mel 31
1.006 2.922 8 3 2 0.225699 -0.127635 -0.098064 63 88 8 32 66 44
```

```
& 1  1 1

!370ncmp.pic     -mel 32
1.067 6.824 15 9 8 0.0741798 -0.0569789 -0.0172009 163 80 13 54 59 29
& 0 1  1

!373ncmp.pic    -mel 33
1.070 3.933 15 8 5 0.0943664 -0.0547818 -0.0395846 193 84 9 55 77 33
& 0 1 1

!379ncmp.pic   -mel 34
1.239 13.017 31 18 13 0.0373689 -0.0233889 -0.01398 169 76 21 60 42 37
& 0 1 1

!380ncmp.pic   -mel 35
1.025 7.680 13 7 5 0.0737474 -0.041749 -0.0319984 201 81 15 61 61 34
& 0 1 1

!390ncmp.pic   -mel 36
1.065 7.208 20 12 9 0.132051 -0.0769104 -0.0551402 147 78 19 56 44 38
& 0 1 1

!392ncmp.pic  -mel 37
1.017 2.868 14 5 3 0.0377666 -0.0116249 -0.0261417 92 78 15 43 45 27
& 1 1 1

!393ncmp.pic   -mel 38
1.426 18.037 10 9 8 0.227762 -0.125936 -0.101826 39 85 17 22 57 34
& 1 1 1

!394ncmp.pic  -mel 39
1.303 17.079 33 17 9 0.0993598 -0.0409916 -0.0583682 126 83 21 53 52 59
& 0 1 1

!396nmp.pic -mel 40
1.134 9.216 12 10 7 0.22399 -0.132576 -0.0914141 34 88 5 20 67 49
& 1 1 1

!397ncmp.pic  -mel 41
1.882 28.123 22 16 15 0.0714966 -0.0360047 -0.0354919 192 73 20 63 42 25
& 0 1 1

!402ncmp.pic  -mel 42
1.939 20.821 33 17 9 0.0844585 0.00212445 -0.0865829 92 78 15 43 45 27
& 0 1 1

!408ncmp.pic  -mel 43
1.734 37.678 25 15 10 0.105073 -0.0489961 -0.0560769 187 76 25 66 39 53
& 0 1 1

!413ncmp.pic  -mel 44
1.588 20.317 19 11 7 0.0467873 -0.0153064 -0.0314809 178 82 13 57 63 36
& 0 1 1

!423ncmp.pic   -mel 45
2.513 37.156 22 16 15 -0.0947313 0.0362838 0.0584475 147 78 19 56 44 38
& 0 1 1

!427ncmp.pic  -mel 46
1.927 21.983 20 15 11 0.190908 -0.113154 -0.0777543 199 84 10 56 78 34
& 1 1 1
```

```
!430ncmp.pic -mel 47
2.120 33.625 25 8 5 0.256002 -0.132912 -0.12309 73 87 5 31 72 39
& 0 1 1

!451ncmp.pic  -mel 48
1.374 25.129 6 3 1 0.236951 -0.120386 -0.116564 54 89 8 30 75 56
& 0 1 1


!453ncmp.pic mel - 49
1.207 13.952 32 23 17 0.101117 -0.0636073 -0.0375097 171 76 22 61 40 42
& 1 1 1

!449ncmp.pic mel - 50
1.101 6.933 19 7 6 0.135948 -0.0620915 -0.0738562 72 85 11 35 63 46
& 1 1 1


!448ncmp.pic mel - 51
1.056 6.406 24 9 4 0.265237 -0.125843 -0.139394 127 85 15 50 62 49
& 0 1 1

!447ncmp.pic mel - 52
1.040 4.521 21 13 9 0.0698387 -0.028629 -0.0412097 100 75 26 50 30 53
& 1 1 1

!446ncmp.pic mel -53
1.423 13.723 17 11 8 0.0980695 -0.0369099 -0.0611596 78 80 20 42 40 51
& 0 1 1


!442ncmp.pic mel -54
1.233 11.957 31 12 8 0.0461114 -0.0233622 -0.0227492 129 82 14 49 54 35
& 0 1 1


!441ncmp.pic mel -55
1.109 10.185 26 18 16 0.0140629 -0.0192942 0.00523138 142 78 14 51 52 27
& 0 1 1


!439ncmp.pic mel -56
1.086 11.702 7 19 13 0.18146 -0.08708 -0.09438 241 79 21 71 54 50
& 0 1 1


!438ncmp.pic mel -57
1.072 10.586 25 14 10 0.10452 -0.0671013 -0.0374185 139 81 17 53 51 40
& 0 1 1



!437ncmp.pic mel -58
1.218 19.313 30 13 6 0.130694 -0.0477818 -0.082912 109 82 18 48 48 48
& 0 1 1


!436ncmp.pic mel -59
1.181 21.203 15 17 11 0.191822 -0.0953228 -0.0964992 200 81 17 62 59 42
& 0 1 1
```

```
!431ncmp.pic mel -60
1.084 15.835 19 7 5 0.180102 -0.090321 -0.0897807 154 82 12 52 62 31
& 0 1 1


!425ncmp.pic mel -61
1.102 11.080 30 15 11 0.164723 -0.0892837 -0.0754392 114 84 12 45 61 40
& 1 1 1


!424ncmp.pic mel -62
1.105 9.108 29 12 7 0.105187 -0.0488049 -0.0563824 104 83 18 47 52 54
& 1 1 1


!407ncmp.pic mel -63
1.055 5.664 13 4 3 0.151061 -0.088169 -0.0628915 79 78 14 40 43 24
& 0 1 1


!406ncmp.pic mel -64
1.450 29.808 20 9 7 0.129347 -0.0608645 -0.0684824 183 77 17 60 51 29
& 0 1 1


!404ncmp.pic mel -65
1.214 13.546 27 20 14 0.068254 -0.0377551 -0.0304989 137 78 17 53 46 36
& 0 0 1


!401ncmp.pic mel -66
1.414 10.134 27 10 7 0.0600894 -0.0245312 -0.0355581 108 82 14 45 53 39
& 0 1 1


!400ncmp.pic mel -67
1.387 20.848 21 16 9 0.221411 -0.104651 -0.11676 184 81 18 61 55 46
& 0 1 1


!399ncmp.pic mel -68
1.150 10.626 29 7 6 0.239975 -0.121867 -0.118108 72 87 4 30 75 39
& 1 1 1


!398ncmp.pic mel -69
1.166 12.681 34 26 16 0.105381 -0.0612049 -0.0441756 190 77 25 66 41 57
& 0 1 1



!301 12170 mel -70
1.709 16.054 28 23 19 0.106254 -0.0577904 -0.0484637 164 79 16 56 52 34
& 0 1 1

!302 23876 mel -71
1.658 42.72 18 7 5 0.156433 -0.0755013 -0.0809315 87 86 14 41 62 52
& 0 1  1

!306 9996 mel -72
```

1.123 13.861 16 7 5 -0.0300952 0.0162374 0.0138578 114 84 17 49 56 53
& 0 1 1

!307 18147 mel -73
1.409 19.286 25 8 6 0.170393 -0.0874531 -0.08294 106 82 10 42 56 27
& 1 1 1

!311 7927 mel -74
1.081 9.456 18 15 12 -0.0812317 0.0240469 0.0571848 97 72 29 50 26 57
& 0 1 1

!312 41967 mel -75
1.164 13.365 17 9 5 0.13557 -0.0738817 -0.0616883 218 77 18 66 52 35
& 0 1 1

!316 46806 mel -76
1.101 11.894 30 22 14 0.146527 -0.0681446 -0.0783824 207 76 25 68 41 51
& 0 1 1

!322 11421 mel -77
1.772 21.919 20 5 3 -0.027129 -0.00618633 0.0333154 93 78 16 44 43 29
& 0 1 1

!326 102310 mel -78
1.420 9.987 17 11 9 -0.0233333 0.00515152 0.0181818 63 72 28 40 23 51
& 1 1 1

!327 29130 mel -79
1.195 7.631 33 19 11 0.161313 -0.0709037 -0.0904095 180 78 26 65 44 61
& 1 1 1

!338 30250 mel -80
1.083 6.765 16 9 4 0.114043 -0.0570584 -0.0569845 88 81 20 45 42 54
& 1 1 1

!341 1208 mel -81
1.255 11.624 35 10 7 0.151232 -0.0747873 -0.0764442 125 86 9 45 71 40
& 0 1 1

!346 41561 mel -82
1.264 9.217 38 15 9 0.0899471 -0.0590829 -0.0308642 137 81 16 51 52 40
& 1 1 1

!351 17372 mel -83
1.199 17.717 10 9 7 0.0536682 -0.0276795 -0.0259887 73 76 21 41 32 37
& 0 1 1

!365 27534 mel -84
2.264 31.872 21 12 8 0.10447 -0.0560487 -0.0484211 75 81 18 40 43 47
& 0 1 1

!367 2846 mel -85
1.154 23.610 30 16 11 0.147373 -0.078599 -0.0687741 150 80 17 55 50 38
& 0 1 1

!371 6774 mel -86
1.077 10.265 25 12 7 0.123074 -0.0724388 -0.0506351 155 82 14 54 60 38
& 0 1 1

```
!375 3209 mel -87
1.106 12.313 34 16 14 0.116197 -0.0688675 -0.0473291 135 72 22 54 34 27
& 1 0 1

!381 17002 mel -88
1.341 8.543 31 15 8 0.144199 -0.0766081 -0.0675909 152 81 20 57 51 53
& 0 1 1

!382 2427 mel -89
1.212 12.231 30 16 11 0.0380793 -0.0193452 -0.0187341 177 78 18 60 48 34
& 0 1 1

!384 7837 mel -90
1.335 18.973 29 19 9 0.0746528 -0.0430556 -0.0315972 103 84 19 47 54 59
& 0 1 1

!387 5932 mel -91
1.327 16.201 27 11 8 0.0776073 -0.0294927 -0.0481145 130 78 22 55 39 49
& 0 1 1

!318ncmp.pic -mel 92
1.249 18.419 10 4 3 0.115542 -0.066803 -0.0487389 91 86 7 38 67 34
& 0 1 1
```

BIBLIOGRAPHY

1. R.J. Friedman, D.S. Rigel, M.K. Silverman, A. W. Kopf and K.A. Mossaert, "Malignant Melanoma in the 1990's: The Continued Importance of Early Detection and The Role of Physician Examination and Self-Examination of the Skin," Ca-A Journal for Clinicians, July/August, pp. 201-226.

2. W.V. Stoecker and R.H. Moss, "Editorial: Digital Imaging in Dermatology, Computerized Medical and Graphics, Vol. 16(3), May-June 1992, pp. 145-150.

3. R.J. Friedman, D.S. Riegel, A. W. Kopf, "Early Detection of Malignant Melanoma: The Role of Physician Examination and Self-Examination of The Skin," Ca-A Cancer Journal for Clinicians, Vol 35(3): pp. 130-151, 1985.

4. National Cancer Institute, What You Need to know about Dysplastic Nevi, NIH Publication 91-3133, Reprinted October 1990.

5. R. Poli, S Cagnoni, R. Livi, G. Coppini, G. Valli, "A Neural Network Expert System for Diagnosing and Treating Hypertension," IEEE Computer , March 1991, pp 64-71.

6. R. P. Lippman, "Review of Neural Networks for Speech Recognition," Neural Computing, Vol. 1(1), 1989, pp 1 -38.

7. T. Kohonen, G. Barna, and P. Christy, "Statistical Pattern Recognition with Neural Networks: Benchmarking Studies," IEEE Annual International Conference on Neural Networks, San Diego, July 1988.

8. R. P. Lippman, "An Introduction to Computing with Neural Nets," IEEE ASSP Mag., vol 4(2), pp. 4-22, Apr. 1987.

9. D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, "A Learning Algorithm for Boltzmann Machines," Cognitive Science, vol. 9, pp 147-160, 1985.

10. C. L. Giles and T. Maxwell, "Learning, Invariance and Generalization in High-Order Networks," Applied Optics, vol. 26, pp 4,972-4,978, Dec. 1987.

11. S. Farlow, Self Organizing Methods in Modelling, Marcel Drekker, 1984.

12. F. H. Glanz and W. T. Miller, "Shape Recognition using a CMAC-Based Learning System," SPIE Proc. Intelligent Robots and Computer Vision, Cambridge, MA, 1987.

13. R. O. Duda and P. E. Hart, Pattern Classification and Scene Analysis, NY: John Wiley and Sons, 1973.

14. W. M. Huang and R. P. Lippman, "Neural Net and Traditional Classifiers," Neural Info. Processing Syst., D. Anderson, ed., pp 387-396, NY: American Institute of Physics, 1988.

15. G. A. Carpenter and S. Grossberg, "ART 2: Self-Organization of Stable Category Recognition Codes for Analog Input Patterns," Applied Optics, vol 26, pp 4194-4930, 1987.

16. R. P. Lippman, "Review of Neural Networks for Speech Recognition," Neural Comp., vol 1 (1), pp. 1-38, 1989.

17. T.J. Sejnowski and C. M. Rosenberg, "Parallel Networks that learn to Pronounce English Text," Complex Systems, vol. 1, pp 145-168, 1987.

18. R. P. Gorman and T. J. Sejnowski, "Analysis of Hidden Units in a Layered Network Trained to Classify Sonar Targets," Neural Networks, vol. 1, pp 75-89, 1988.

19. G. Tesauro and T. J. Sejnowski, "A Neural Network that Learns to Play Backgammon," Neural Information Processing Systems, D. Anderson, ed., pp 794-803, American Institute of Physics, New York, 1988.

20. R.P. Lippman and P. E. Beckman, "Adaptive Neural Net Processing for Signal Detection in Non-Gaussian Noise," Advances in Neural Info. Processing Syst. 1, D. S. Touretzky, ed., San Mateo, CA: Morgan Kaufmann, 1989.

21. S. J. Hanson and D. J. Burr, "Knowledge Representation in Connectionist Networks," Tech. Rep., Bell Communications Research, Morristown, New Jersey, Feb. 1987.

22. I. D. Longstaff and J. F. Cross, "A Pattern Recognition Approach to Understanding the Multi-Layer Perceptron," Memo. 3,936, Royal Signals and Radar Establishment, July 1986.

23. A. Lapedes and R. Farber, "How Neural Nets Work," Neural Information Processing Systems, D. Anderson, ed., pp 442-456, American Institute of Physics, New York, 1988.

24. A. Wieland and R. Leighton, "Geometric Analysis of Neural Network Capabilities," IEEE 1st International Conf. on Neural Networks, San Diego, CA, pp 111-385, June 1987.

25. G. Cybenko, "Approximation by Superpositions of a Sigmoidal Function," Mathematics of Control, Signals, and Systems, 2(4),1989.

26. D. J. Burr, "Experiments on Neural Net Recognition of Spoken and Written Text," IEEE Transactions on Acoustics, Speech and Signal Processing, 36:pp. 1162-1168, 1988.

27. S. J. Hanson and L.Y. Pratt, "Some Comparisons of Constraints for Minimal Network Construction with BackPropagation," Advances in Neural Information Processing Systems 1, D.S. Touretzky, ed., San Mateo, CA: Morgan Kaufman, 1989.

28. J. E. Golston, R.H. Moss, W. V. Stoecker, "Boundary Detection in Skin Tumor Images: An Overall Approach and a Radial Search Algorithm," Pattern Recognition, Vol. 23, no. 11, 1990, pp. 1235-1247.

29. M. Moganti, F. Ercal, W. V. Stoecker, and R. H. Moss, "Boundary Detection in Skin Tumor Images," To appear in IEEE Transactions on Medical Imaging.

30. S. E. Umbaugh, Computer Metrics in medicine: Color metrics and Image Segmentation Methods for Skin Cancer Diagnosis, Ph. D. Dissertation, Elect. Eng. Dept., University of Missouri-Rolla. UMI Dissertation Services, Ann Arbor, Michigan, 1990.

31. M. Celenk, "Color Image Segmentation by Clustering," IEE Proceedings, Vol. 138, no. 5, September 1991, pp. 368-376.