A Probabilistic Neural Network Framework for Detection of Malignant Melanoma

Mads Hintz-Madsen, Lars Kai Hansen[1] and Jan Larsen
CONNECT, Dept. of Mathematical Modelling, Build. 321,
Technical University of Denmark, DK-2800 Lyngby, Denmark,
Phone: (+45) 4525 3885, Fax: (+45) 4587 2599,
Email: mhm, lkhansen, jl@imm.dtu.dk

Krzysztof T. Drzewiecki
Dept. of Plastic Surgery S, National University Hospital,
Blegdamsvej 9, DK-2100 Copenhagen, Denmark,
Phone: (+45) 3545 3030

[1]Corresponding author. This research is supported by the Danish Research Councils for the Natural and Technical Sciences through the Danish Computational Neural Network Center and the THOR Center for Neuroinformatics.

Contents

1 INTRODUCTION

The work reported in this chapter concerns the classification of dermatoscopic images of skin lesions. The overarching goals of the work are:

Develop an objective and cost-efficient tool for classification of skin lesions

This involves extracting relevant information from dermatoscopic images in the form of dermatoscopic features and designing reliable classifiers.

Gain insight into the importance of dermatoscopic features

The importance of dermatoscopic features is still very much a matter of research. Any additional insight into this area is desirable.

Develop a probabilistic neural classifier design framework

In order to obtain reliable classification systems based on neural networks, a principled probabilistic approach will be followed.

Hence, the work should be of interest to both the dermatological and engineering communities.

1.1 Malignant melanoma

Malignant melanoma is the deadliest form of skin cancer and arises from cancerous growth in pigmented skin lesions. The cancer can be removed by a fairly simple surgical incision if it has not entered the blood stream. It is thus vital that the cancer is detected at an early stage in order to increase the probability of a complete recovery. Skin lesions may in this context be grouped into three classes:

- *Benign nevi* is a common name for all healthy skin lesions. These have no increased risk of developing cancer.
- *Atypical nevi* are also healthy skin lesions but have an increased risk of developing into cancerous lesions. The special type of atypical nevi called *dysplastic nevi* have the highest risk and are, thus, often referred to as precursors of malignant melanoma.
- *Malignant melanoma* are as already mentioned cancerous skin lesions.

When a dermatologist inspects a skin lesion and finds it suspect, the dermatologist will remove the skin lesion and a biopsy is performed in order to determine the exact type of skin lesion. If the lesion is found to be malignant, a larger part of the surrounding skin will be removed depending on the degree of malignancy. If a lesion is not considered to be suspect, it is usually not removed unless there is some cosmetic reason to do so.

It is not an easy task for dermatologists visually to determine whether a skin lesion is or might be malignant, though. A study at *Karolinske Hospital, Stockholm, Sweden* has shown that a dermatologists with less than 1 year of experience detects 31% of the melanoma cases they are presented with while dermatologists with more than 10 years of experience are able to detect 63% [1]. Another study shows that experienced dermatologist are capable of detecting 75% of cancerous skin lesions [2].

Malignant melanoma is usually only seen in Caucasians.

1.2 Evolution of malignant melanoma

The incidence of malignant melanoma in Denmark has increased 5- to 6-fold from 1942 to 1982 while the mortality rate has been doubled from 1955 to 1982 [3]. Currently, approximately 800 cases of malignant melanoma are reported in Denmark every year. In Germany $9000 - 10000$ new cases are expected every year with an annual increase of $5 - 10\%$ [4].

Due to the rather steep increase in the number of reported malignant melanoma cases, it is becoming increasingly important to develop methods capable of diagnosing malignant melanoma that are simple, objective and preferably non-invasive. Today the only accurate diagnostic technique is a biopsy and a histological analysis of the skin tissue sample. This is an expensive procedure as well as an uncomfortable experience for the patient. For patients with many skin lesions or *dysplastic nevus syndrome*[1], this is clearly not a feasible diagnostic technique. Contributing to the problem is the increasing awareness of skin cancer among the general public. People are consulting dermatologists more often which again calls for a simple and accurate diagnostic technique.

1.3 Image acquisition techniques

1.3.1 Traditional imaging

In larger dermatological clinics, records of the patients skin lesions are kept in form of a diagnosis and one or more traditional photographs of the lesion. Some patients may be predisposed to melanoma due to, e.g., cancer in the family or dysplastic nevus syndrome. These patients will often be regularly checked in order to detect any changes in their skin lesions. Photographs taken at each check-up are compared and any change is an indication of a possible malignancy. In this case, the lesion is removed and a biopsy performed.

It is mainly for this monitoring over time that traditional imaging is used today. An example of a traditional photograph is shown in figure 1.

[1] People with *dysplastic nevus syndrome* have multiple dysplastic nevi - often dozens or even hundreds.

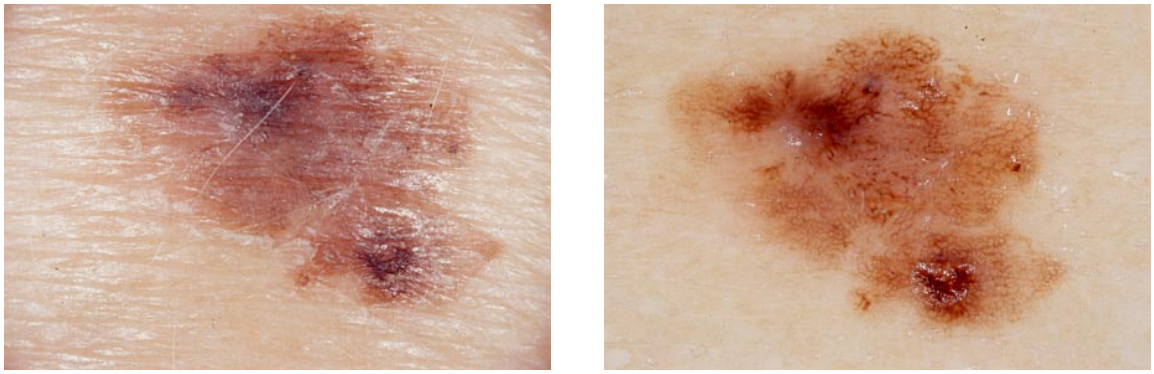

Figure 1: Example of pigmented skin lesion. Left: Traditional imaging technique. Right: Dermatoscopy imaging technique.

1.3.2 Dermatoscopic imaging

Since traditional imaging is just a recording of what the human eye sees, it does not reveal any information unavailable to the eye. *Dermatoscopy* also known as *epiluminescence microscopy*, on the other hand, is an imaging technique that provides a more direct link between biology and distinct visual characteristics.

Dermatoscopy is a non-invasive imaging technique that renders the *stratum corneum*[2] translucent and makes subsurface structures of the skin visible. The technique is fairly simple and involves removing reflections from the skin surface. This is done by applying immersion oil onto the skin lesion and pressing a glass plate with the same reflection index as the stratum corneum onto the lesion. The oil ensures that small cavities between the skin and the glass plate are filled in order to reduce reflections. With a strong lightsource, usually a halogen lamp, it is now possible to see skin structures below the skin surface. Usually the glass plate and lightsource are integrated into devices like a *dermatoscope* or a *dermatoscopic camera*. Both of these have lenses allowing a 10x magnification of pigmented skin lesions. In figure 1 an example of a skin lesion, recorded by the dermatoscopic imaging technique, is shown.

Although this imaging technique is not new, it is only in the last decade that the technique has been thoroughly investigated, especially in Western Europe [5]. It is still, though, not a widely used technique primarily due to the lack of formal training in evaluating and understanding the visual characteristics in the images. Some of these characteristics will be briefly described in the next section.

A few studies concerning processing and analysis of digital dermatoscopic images have been published. In [6] and [7], results of color segmentation techniques based on fuzzy c-means clustering are shown. Preliminary results using a minimum-distance classifier for discriminating between benign nevi, dysplastic nevi and malignant melanoma are presented in [8]. Based on features describing various properties including shape and color, they were able to classify 56% of skin lesions in a test set correctly.

[2]The top layer of the skin.

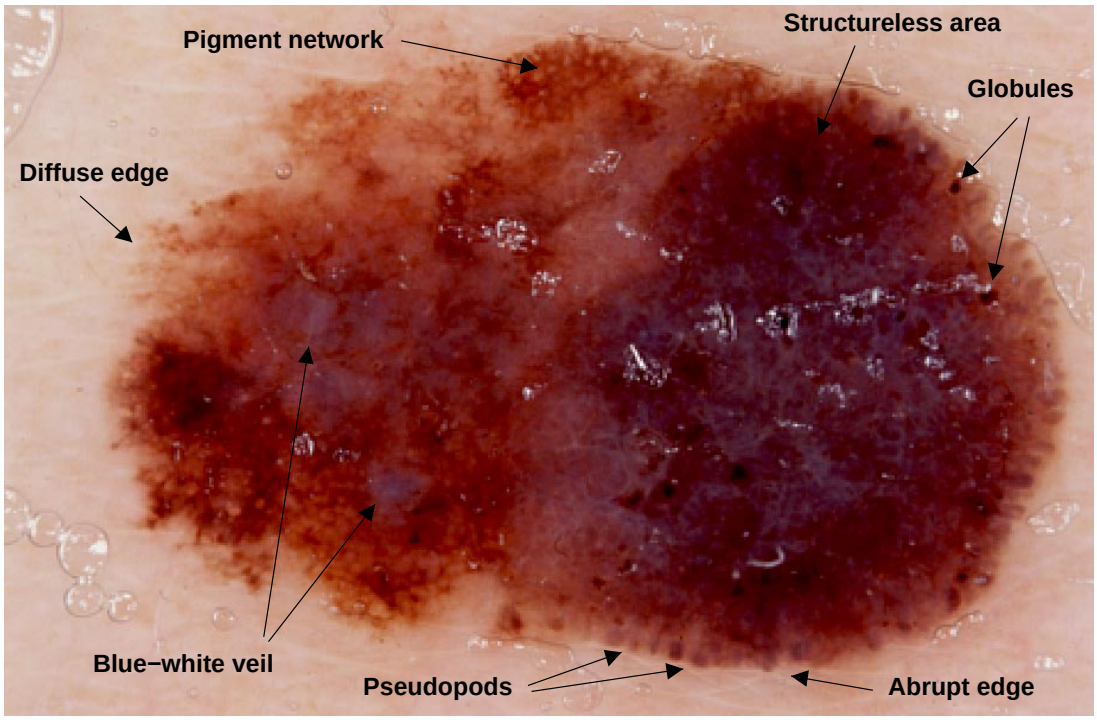

Figure 2: Pigmented skin lesion with several dermatoscopic features.

1.4 Dermatoscopic features

The dermatoscopic imaging technique produces images that are quite different from traditional images. Several visual characteristics have been defined and analyzed in recent studies, e.g., [9], [10] and [11]. These visual characteristics will be called *dermatoscopic features* or just *features* for short.

Table 1 lists the most important dermatoscopic features together with a short description. The features all describe specific biological behavior, see, e.g., [10] for a more detailed description. In figure 2 and several dermatoscopic features are shown on a pigmented skin lesion.

As can be seen in, e.g., figure 2, there is one prominent artifact due to the use of immersion oil. Small air bubbles occur in the oil layer and appear as small white circles or ellipses. This artifact can be avoided if the oil is carefully applied. Usually the area occupied by air bubbles is very small but important features like, e.g., black dots or pseudopods may be obscured by air bubbles.

Table 1: Definition of dermatoscopic features

Feature	Description
Asymmetry	An asymmetric shape is the result of different local growth rates. This indicates malignancy. Asymmetry may be defined in numerous ways, though. In section 2.4.1, one such definition is presented.
Edge abruptness	A sharp abrupt edge suggests melanoma while a gradual fading of the pigmentation indicates a benign lesion.
Color distribution	Six different colors may be observed: Light-brown, dark-brown, white, red, blue and black. A large number of colors present indicates melanoma.
Pigment network	Areas with honeycomb-like pigmentation. A regular network usually indicates a benign lesion. A network with varying mesh size suggests an atypical/dysplastic nevus or a melanoma.
Structureless area	Areas with pigmentation but without any visible network. Unevenly distributed areas indicate melanoma.
Globules	Nests with a diameter of more than $0.1mm$ of heavily pigmented melanocytic cells. These may be brown or black. If evenly distributed, it indicates a benign lesion.
Black dots	Heavily pigmented melanocytic cells with a diameter less than $0.1mm$. If located close to the perimeter, it suggests an atypical lesion or a melanoma.
Pseudopods	Large "rain-drop" shaped melanoma nests located at the edge of the lesion. A very strong indicator of malignant melanoma.
Radial streaming	Radial growth of melanoma. Looks like streaks. Very indicative of malignant melanoma.
Blue-white veil	Areas with a blue-white shade of color. Indicates melanocytic cells located deep in the skin. An indicator of melanoma.
Depigmentation	Loss of pigmentation. An indicator of melanoma.

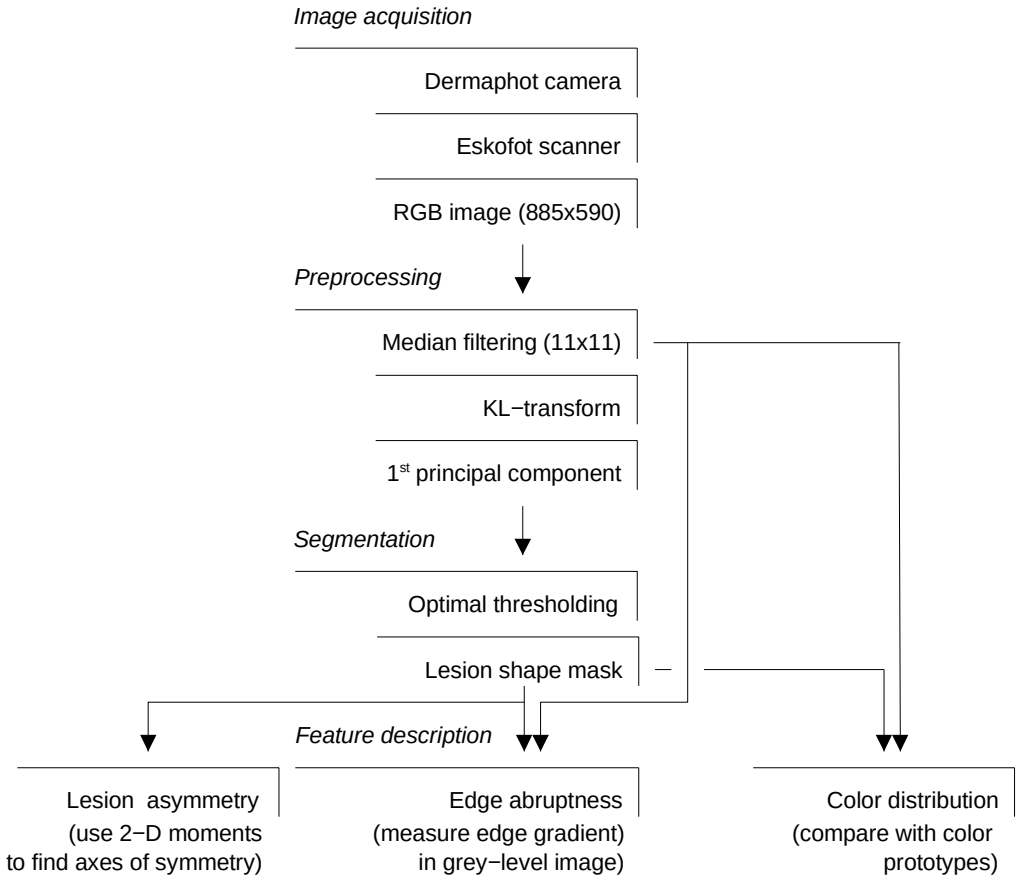

Figure 3: Feature extraction flowchart showing the four main processing blocks, image acquisition, preprocessing, segmentation and feature description.

2 FEATURE EXTRACTION IN DERMATOSCOPIC IMAGES

In the previous section, dermatoscopic images and features were introduced. In this section, we will describe the image processing techniques used in order to extract and describe dermatoscopic features.

In figure 3, a flowchart describing the feature extraction process is shown. The four main blocks, image acquisition, preprocessing, segmentation and dermatoscopic feature description, are described in the next sections.

2.1 Image acquisition

All dermatoscopic images used in this work are acquired at *Rigshospitalet, Copenhagen, Denmark* using a *Dermaphot* camera (*Heine Optotechnik*).

The images are developed as slides and digitalized with a resolution of 1270 dots per inch and 24 bit color[3] using an *Eskoscan 2540* color scanner (*Eskofot*). The image resolution has later digitally been

[3] 8 bit for each of the color channels red, green and blue.

reduced by a factor 2 in order to limit the computational resources needed for processing the images, thus reducing the size of each image to 885x590.

2.2 Image preprocessing

The first step in the feature extraction process is preprocessing of images with the purpose of reducing noise and facilitating image segmentation by using median filtering and the Karhunen-Loéve transform.

Now, let us first define a grey-level image of size MxN as a sequence of numbers,

$$z(m,n), \quad 1 \leq m \leq M, 1 \leq n \leq N, \tag{1}$$

where $z(m,n)$ is the luminance of pixel (m,n). If we are dealing with an 8-bit grey-level image, then each element, $z(m,n)$, will be an integer in the interval $[0;255]$. In any processing of 8-bit images, we will abandon the integer restriction and process the image in a floating point representation in order to minimize quantization effects.

Next, we define a color image of size MxN as 3 sequences, $r(m,n), g(m,n)$ and $b(m,n)$, with $r(m,n)$ representing the red color component, $g(m,n)$ the green color component and $b(m,n)$ the blue color component. The individual color components are typically representated by 8-bit but again any processing will be done using floating point precision.

2.2.1 Median filtering

As noted in section 1.4, the immersion oil used in the dermatoscopic imaging technique may produce small air bubbles manifestating themselves as small white ellipses, lines or dots. This artifact can be considered as impulsive noise and may thus be reduced using a median filter given by

$$z_{\text{med}}(m,n) = \text{median}\{z(m-k,n-l)| - \frac{N_{\text{med}}-1}{2} \leq k,l \leq \frac{N_{\text{med}}-1}{2} \\ \wedge 1 \leq m-k \leq M \wedge 1 \leq n-l \leq N\}, \tag{2}$$

where N_{med} is odd[4] and indicates the size of the two-dimensional median filter. Note that we only consider a square median filter kernel. We may in fact consider any shape of filter kernel if desirable. Equation (2) is valid for a grey-level image. When working with color images, one should apply the same median filter to all 3 color components.

[4]If the median kernel size is even, there will be two middle values. One could then define the median as the mean of these two values.

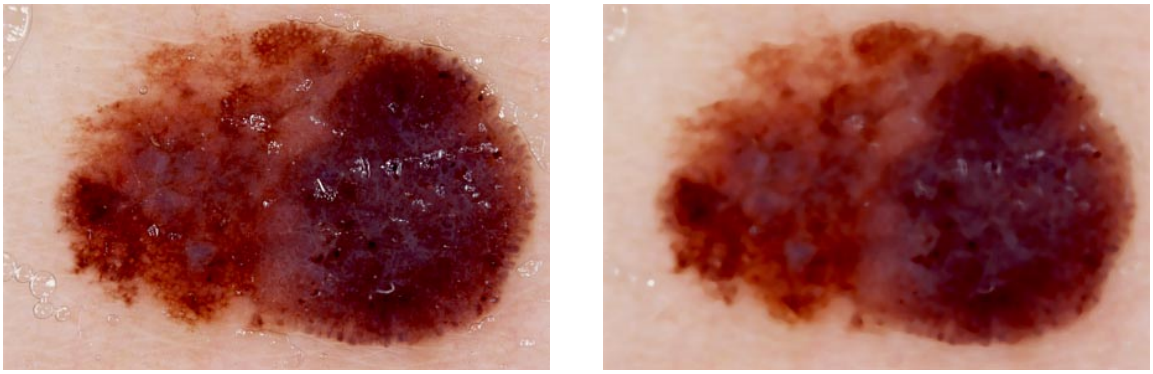

Figure 4: The effect of filtering a 885x590 dermatoscopic image with a 11x11 median filter. Left: Original image. Right: Filtered image. Notice how the air bubble artifacts have been reduced, especially around the lesion edge in the upper right hand corner.

Skin lesion specific comments

The main purpose of filtering dermatoscopic images is to reduce localized reflection artifacts while at the same time preserving edges. In figure 4, the results of applying a 11x11 median filter to a dermatoscopic image is shown. This kernel size is used for all median filtering in this work.

2.2.2 Karhunen-Loève transform

The next preprocessing stage aims at facilitating the segmentation process by enhancing the edges in the image. For this purpose, we will consider the *Karhunen-Loève* (KL) transform also known as the *Hotelling* transform or the method of principal components [12], [13].

The KL transform is a linear transformation that uncorrelates the input variables by employing an orthonormal basis found by an eigenvalue decomposition of the sample covariance matrix for the input variables.

In image processing applications, the KL transformation is often applied to the 2-D image domain. Here we will apply the transformation to the 3-D color space spanned by $r(m,n)$, $g(m,n)$ and $b(m,n)$.

Now, let us define the following $3 \times MN$ matrix containing all pixels from the 3 color channels,

$$\mathbf{V} = \begin{bmatrix} r(1,1) & r(1,2) & \dots & r(1,N) & r(2,1) & \dots & r(M,N) \\ g(1,1) & g(1,2) & \dots & g(1,N) & g(2,1) & \dots & g(M,N) \\ b(1,1) & b(1,2) & \dots & b(1,N) & b(2,1) & \dots & b(M,N) \end{bmatrix}, \tag{3}$$

where we view $[r(m,n)\ g(m,n)\ b(m,n)]^{\mathrm{T}}$ as a sample of a stochastic variable.

Let $\bar{\mathbf{v}}$ contain the sample mean of the 3 color components,

$$\bar{\mathbf{v}} = \frac{1}{MN} \sum_{m=1}^{M} \sum_{n=1}^{N} \begin{bmatrix} r(m,n) \\ g(m,n) \\ b(m,n) \end{bmatrix}. \tag{4}$$

The sample covariance matrix is now given by

$$\mathbf{C} = \frac{1}{MN} \mathbf{V}\mathbf{V}^{\mathrm{T}} - \bar{\mathbf{v}}\bar{\mathbf{v}}^{\mathrm{T}}, \tag{5}$$

that can be eigenvalue decomposed, so that

$$\mathbf{C} = \mathbf{E}\mathbf{\Lambda}\mathbf{E}^{\mathrm{T}}, \tag{6}$$

where $E = [\mathbf{e}_1\ \mathbf{e}_2\ \mathbf{e}_3]$ is a matrix containing the eigenvectors of $\mathbf{C}$ and $\mathbf{\Lambda}$ a diagonal matrix containing the corresponding eigenvalues of $\mathbf{C}$ in decreasing order: $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq 0$.

The KL transformation is now defined as

$$\mathbf{z} = \mathbf{E}^{\mathrm{T}} (\mathbf{v} - \bar{\mathbf{v}}) \tag{7}$$

where $\mathbf{v}$ is a column vector in $\mathbf{V}$ and $\mathbf{z}$ contains what is known as the *principal components*.

Due to the decreasing ordering of the eigenvalues and the corresponding eigenvectors, the first principal component will contain the maximum variance. In fact, no other linear transformation using unit length basis vectors can produce components with a variance larger than λ_1 [14].

Skin lesion specific comments

For median filtered dermatoscopic images, the first principal component will typically account for more than 95% of the total variance. Since most variation occur at edges between regions with similar luminance levels, the first principal component is a natural choice for segmentation. Another study also shows that the Karhunen-Loève transform is appropriate for segmenting dermatoscopic images [6].

2.3 Image segmentation

The next step in the feature extraction process is *image segmentation*. The main goal is to divide an image into regions of interests from which appropriate features can be extracted. Here, we will consider a complete segmentation that divides the entire image into disjoint regions. Denoting the image, $\mathcal{R}$, and the N regions, $\mathcal{R}_i,\ i = 1, 2, \ldots, N$, this may be formalized as

$$\mathcal{R} = \bigcup_{i=1}^{N} \mathcal{R}_i, \qquad \mathcal{R}_i \cap \mathcal{R}_j = \emptyset,\ i \neq j. \tag{8}$$

The regions are usually constructed so that they are homogeneous with respect to some chosen property like, e.g., luminance, color or context. We will now consider the case where the aim is to group pixels containing the same approximate luminance level.

2.3.1 Optimal thresholding

Thresholding is a very simple segmentation method based on using thresholds on the luminance level of pixels in order to determine what region a pixel belongs to. Denoting the non-negative luminance of a pixel, $z(m,n)$, a thresholding process using $N-1$ thresholds to divide an image into N regions may be written as

$$z(m,n) \in \begin{cases} \mathcal{R}_1 & \text{if } z(m,n) < T_1 \\ \mathcal{R}_2 & \text{if } T_1 \leq z(m,n) < T_2 \\ \vdots & \vdots \\ \mathcal{R}_i & \text{if } T_{i-1} \leq z(m,n) < T_i \\ \vdots & \vdots \\ \mathcal{R}_N & \text{if } T_{N-1} \leq z(m,n) \end{cases}, \tag{9}$$

where T_i is the threshold separating pixels in region $\mathcal{R}_i$ from pixels in region $\mathcal{R}_{i+1}$.

Let us consider the luminance level, $z(m,n)$, to be a sample of a stochastic variable, z, and let the conditional luminance probability distribution be denoted by $p(z|\mathcal{R}_i)$ and the prior region probability by $P(\mathcal{R}_i)$. Assuming we know $p(z|\mathcal{R}_i)$ and $P(\mathcal{R}_i)$, we may view the problem of selecting the thresholds as a classification problem and use Bayesian decision theory to minimize the probability of misclassifying a pixel.

Let us now assume that the conditional luminance probability distributions, $p(z|\mathcal{R}_i)$, are Gaussian with mean $\mu_{\mathcal{R}_i}$ and equal variance $\sigma^2_{\mathcal{R}_i} = \sigma^2$. We thus obtain the following closed-form solution for the optimal thresholds,

$$T_i = \frac{\mu_{\mathcal{R}_i} + \mu_{\mathcal{R}_{i+1}}}{2} + \frac{\sigma^2}{\mu_{\mathcal{R}_i} - \mu_{\mathcal{R}_{i+1}}} \log \frac{P(\mathcal{R}_{i+1})}{P(\mathcal{R}_i)}, \tag{10}$$

where $i = 1, 2, \ldots, N-1$. Assuming the prior probabilities, $P(\mathcal{R}_i)$, are equal, equation (10) reduces to

$$T_i = \frac{\mu_{\mathcal{R}_i} + \mu_{\mathcal{R}_{i+1}}}{2}. \tag{11}$$

A simple iterative scheme based on equation (11) for estimating the $N-1$ optimal thresholds and the N luminance means is [15]

1. Initialize thresholds, so that $T_1 < T_2 < \ldots < T_{N-1}$.

2. At time step t, compute the luminance region means

$$\mu_{\mathcal{R}_i}^{(t)} = \frac{\sum_{(m,n)\in\mathcal{R}_i^{(t)}} z(m,n)}{N_{\mathcal{R}_i}^{(t)}}, \tag{12}$$

   where $N_{\mathcal{R}_i}^{(t)}$ is the number of pixels in region $\mathcal{R}_i$ at time step t and $i = 1, 2, \ldots, N$.

3. The thresholds at time step $t+1$ are now computed as

$$T_i^{(t+1)} = \frac{\mu_{\mathcal{R}_i}^{(t)} + \mu_{\mathcal{R}_{i+1}}^{(t)}}{2}, \tag{13}$$

   where $i = 1, 2, \ldots, N-1$.

4. If $T_i^{(t+1)} = T_i^{(t)}$ for all $i = 1, 2, \ldots, N-1$, then stop; otherwise return to step 2.

Skin lesion specific comments

All dermatoscopic images in this work have been segmented by the optimal thresholding algorithm using 2 thresholds. A typical first principal component of a median filtered dermatoscopic image consists of a very light background and a dark skin lesion with even darker areas inside. These 3 regions are usually fairly homogeneous making the assumption of Gaussian luminance probability distributions a sound one. The assumptions of equal variances, $\sigma_{\mathcal{R}_i}^2$ and equal priors, $P(\mathcal{R}_i)$, are usually not warranted. Nevertheless, the algorithm provides good results using dermatoscopic images.

Note, the main purpose of segmentation in this application is to find a lesion shape mask defining the edge location of the lesion. Thus, we are only interested in the threshold separating the light skin background and the darker skin lesion. In some cases, the segmentation produces several skin lesion candidates due to other small non-lesion objects. Usually the largest object is the skin lesion and is thus selected for further processing.

In figure 5, the results of using the optimal thresholding algorithm on a dermatoscopic image using 2 thresholds to separate 3 regions are shown. Note, the similar shape of the sample histogram and the estimated histogram indicating the usability of the optimal thresholding algorithm in the context of dermatoscopic images.

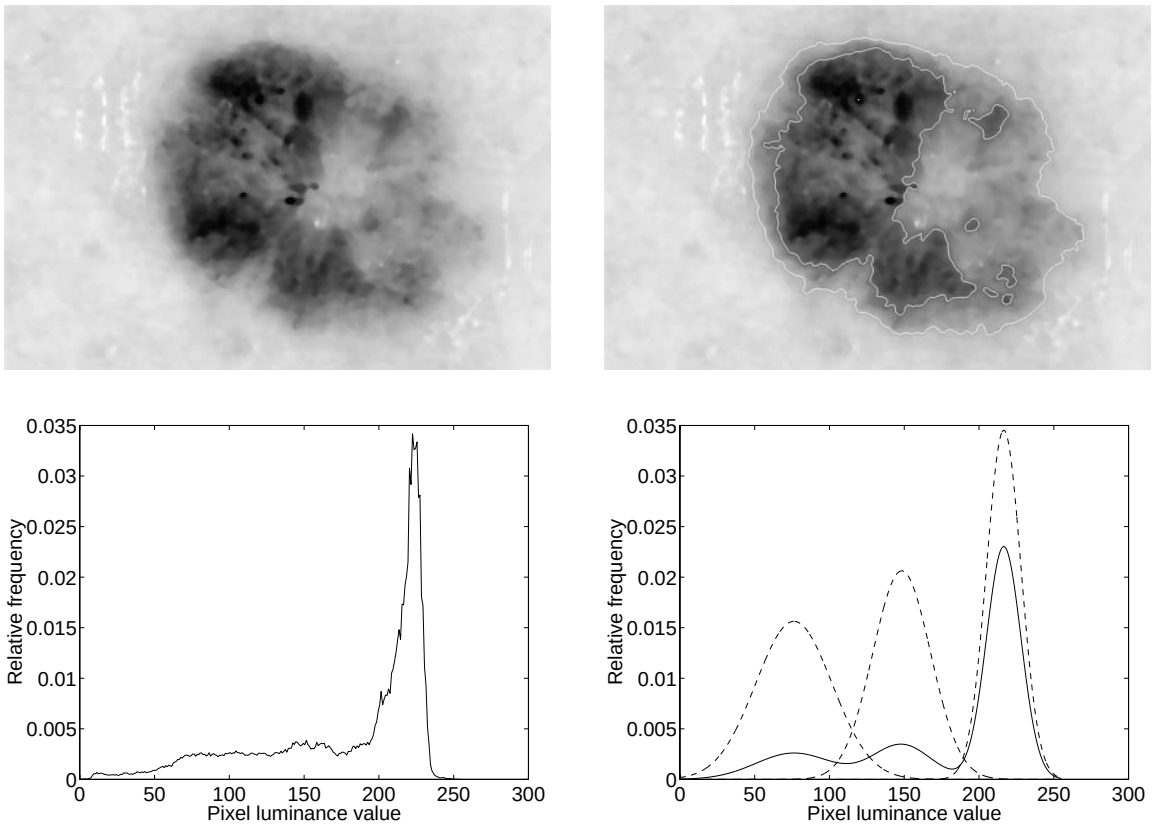

Figure 5: Example of results using the optimal thresholding algorithm on the first principal component of a median filtered dermatoscopic image. Upper left: Median filtered first principal component. Upper right: The segmentation result using 2 thresholds to separate 3 regions. The solid white lines indicate region borders. Lower left: The sample histogram of the upper left image. Lower right: Estimated histogram. The dashed lines show the luminance probability densities, $\hat{p}(z|\mathcal{R}_i)$, estimated by the optimal thresholding algorithm. The solid line shows the estimated histogram computed by assuming that the prior probabilities of the 3 regions are $1/6, 1/6$ and $4/6$ from left to right. Note, that the overall shape of the estimated histogram matches the sample histogram fairly well.

2.4 Dermatoscopic feature description

The final step in the feature extraction process is the actual extraction and description of features. We will in this section present methods for describing the following skin lesion properties:

- Asymmetry of the lesion border.
- Transition of the pigmentation from the skin lesion to the surrounding skin.
- Color distribution of the skin lesion including blue-white veil.

2.4.1 Asymmetry

An asymmetric skin lesion shape is the result of different local growth rates and may indicate malignancy.

In order to measure asymmetry, we will first look at 2-D moments and how these may be used for describing certain geometrical properties of an object or a region in an image.

Moments

Moment representations interpret a normalized grey level image function, $z(x, y)$, as a probability density function of a 2-D stochastic variable. Properties of this variable may thus be described by 2-D moments [16]. For a digital image, $z(m, n)$, the *moment of order* $(p + q)$ is given by

$$m_{pq} = \sum_{m=1}^{M} \sum_{n=1}^{N} m^p n^q z(m, n). \tag{14}$$

Translation invariant moments are obtained by considering the *centralized moments*

$$m_{pq}^c = \sum_{m=1}^{M} \sum_{n=1}^{N} (m - m_c)^p (n - n_c)^q z(m, n), \tag{15}$$

where (m_c, n_c) is the *center of mass* given by $m_c = \frac{m_{10}}{m_{00}}$, $n_c = \frac{m_{01}}{m_{00}}$.

We will now in the following consider the case where $z(m, n)$ is binary and represents a region, $\mathcal{R}$, so that $z(m, n) = 1$ if $(m, n) \in \mathcal{R}$, otherwise $z(m, n) = 0$. This could, e.g., be the result of a segmentation process.

The moment of inertia for a binary object or region, $\mathcal{R}$, w.r.t. an axis through the center of mass with an angle θ as shown in figure 6 is defined as [17]

$$I(\theta) = \sum_{(m,n) \in \mathcal{R}} \sum D_\theta^2(m, n) \tag{16}$$

$$= \sum_{(m,n) \in \mathcal{R}} \sum \left[-(m - m_c) \sin\theta + (n - n_c) \cos\theta \right]^2, \tag{17}$$

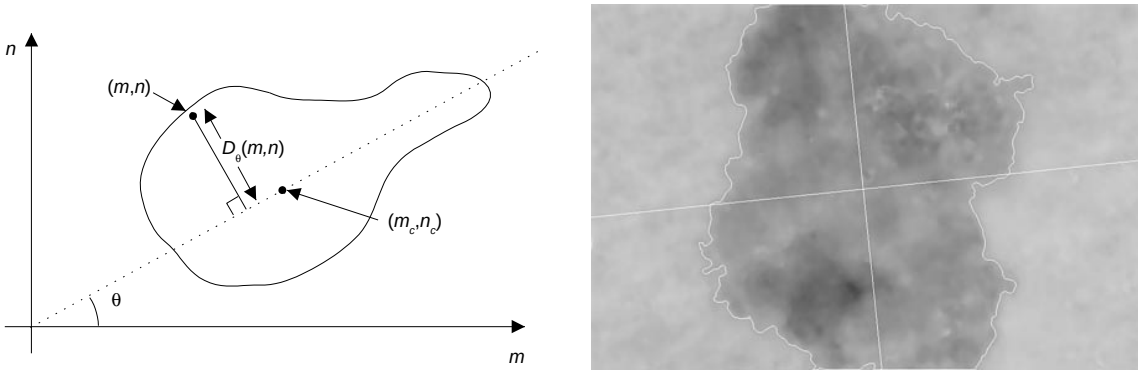

Figure 6: Left: The orientation angle, θ_o, of an object is defined as the angle of the axis through the center of mass, (m_c, n_c), that minimizes the moment of inertia, $I(\theta) = \sum_{m=1}^{M} \sum_{n=1}^{N} D_\theta^2(m,n) z(m,n)$. Right: Skin lesion showing the edge of the lesion and the two principal axes used for calculating asymmetry. These axes define directions of least and largest moments of inertia. The two asymmetry indexes for this lesion are 0.14, respectively. Note, that this lesion is larger than the field of view of the camera. Only very large lesions, where the calculation of asymmetry can not be justified, have been omitted from the data set.

where $D_\theta(m,n)$ is found by translating the object so that its center of mass coincides with the center of origo of the coordinate system and by rotating[5] the object clockwise by the angle θ so that the n-coordinate of the translated and rotated point (m,n) equals the desired distance $D_\theta(m,n)$.

The orientation of an object is defined as the angle of the axis through the center of mass that results in the least moment of inertia [17]. To obtain this angle, we compute the derivative of equation (17) and set it to zero,

$$\frac{\partial I(\theta)}{\partial \theta} = 0 \;\Rightarrow\; \theta_o = \frac{1}{2} \tan^{-1} \left[\frac{2m_{11}^c}{m_{20}^c - m_{02}^c} \right]. \tag{18}$$

The axis through the center of mass defined by θ_o is also known as a *principal axis*. We will refer to this as the *major axis*. All objects have two principal axes[6] where the second principal axis is defined by the angle yielding the largest moment of inertia. This will be referred to as the *minor axis*. The principal axes are orthogonal and will in the next section be used for calculating asymmetry.

In figure 6, an example of a skin lesion and its two principal axes are shown.

Measuring asymmetry

The principal axes found in the previous section will now be used as axes of symmetry. That is, we will measure how asymmetric the object is with respect to these two axes. This can be done by folding the

[5] Rotation of a point (m,n) clockwise by the angle θ is given by: $(m_r, n_r) = (m\cos\theta + n\sin\theta, -m\sin\theta + n\cos\theta)$.

[6] Note, a circle has an infinite number of principal axes due to its rotational symmetry.

object about its principal axes and measure the area of the non-overlapping regions relative to the entire object area. Thus, for each principal axis, we define a measure of asymmetry as

$$S_i = \frac{\Delta A_i}{A}, \tag{19}$$

where $i = 1, 2$ indicates the principal axis, ΔA_i is the corresponding non-overlapping area of the folded object and A is the area of the entire region. For an object completely symmetric about the $i'th$ principal axis, S_i is zero while complete asymmetry yields an asymmetry measure of 1.

Skin lesion specific comments

Several skin lesions included in this work are larger than the field of view of the camera. That is, the entire lesion in not visible in the digitized image. This will introduce an uncertainty in the location of the principal axes and subsequently in the asymmetry measures. See the example in figure 6.

Due to the rather limited amount of data available, these have nevertheless been included. Some severe cases, where the calculation of asymmetry could not be justified, have been removed from the data set, though. One could also choose not to compute the asymmetry measures in these cases and subsequently treat them as missing values. Several techniques for dealing with missing values exist, see, e.g., [18] for an overview.

2.4.2 Edge abruptness

An important feature is the transition of the pigmentation between the skin lesion and the surrounding skin. A sharp abrupt edge suggests malignancy while a gradual fading of the pigmentation indicates a benign lesion.

In order to measure the edge abruptness, let us first estimate the gradient of a grey-level image.

Image gradient estimation

In a digital image, $z(m, n)$, the gradient magnitude, $g(m, n)$ and gradient direction, $\theta_g(m, n)$, is defined by [16]

$$g(m,n) = \sqrt{g_1^2(m,n) + g_2^2(m,n)}, \quad \theta_g(m,n) = \tan^{-1}\left(\frac{g_2(m,n)}{g_1(m,n)}\right), \tag{20}$$

where $g_1(m, n)$ and $g_2(m, n)$ are the difference approximations to the partial derivatives in the m and n direction, respectively,

$$g_1(m,n) = \sum_i \sum_j h_1(-i,-j) z(m+i, m+j) \tag{21}$$

$$g_2(m,n) = \sum_i \sum_j h_2(-i,-j) z(m+i, m+j). \tag{22}$$

$g_1(m,n)$ and $g_2(m,n)$ are expressed as convolutions between the image and *gradient operators* denoted by $h_1(i,j)$ and $h_2(i,j)$, $-(N_h-1)/2 \leq i,j \leq (N_h-1)/2$, where N_h is odd and indicates the size of the gradient operators.

Several gradient operators have been suggested, see, e.g., [17]. Here we will use the Sobel gradient operator defined by

$$\mathbf{H}_1 = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \quad \mathbf{H}_2 = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}. \tag{23}$$

We will in the following denote the gradient magnitude estimation of a grey-level digital image, $z(m,n)$, using the Sobel gradient operators by $g(m,n) = \text{grad}[z(m,n)]$.

Measuring edge abruptness

Let us consider the luminance component of a color image given by

$$z(m,n) = \frac{1}{3}[r(m,n) + g(m,n) + b(m,n)], \tag{24}$$

which is just an equally weighted sum of the three color components.

We may now estimate the gradient magnitude of the intensity component by computing $g(m,n) = \text{grad}[z(m,n)]$.

If we sample the gradient magnitude, $g(m,n)$, along the edge of the skin lesion, we obtain a set of gradient magnitude values,

$$e(k) = g(m(k), n(k)), \quad k = 0, 1, \ldots, K-1, \tag{25}$$

where K is the total number of edge samples and $(m(k), n(k))$ the coordinates of the $k'th$ edge pixel.

This set of values describes the transition between the lesion and the skin background in each edge point. In order to describe the general transition or abruptness, we use the sample mean and variance of the gradient magnitude values $e(k)$,

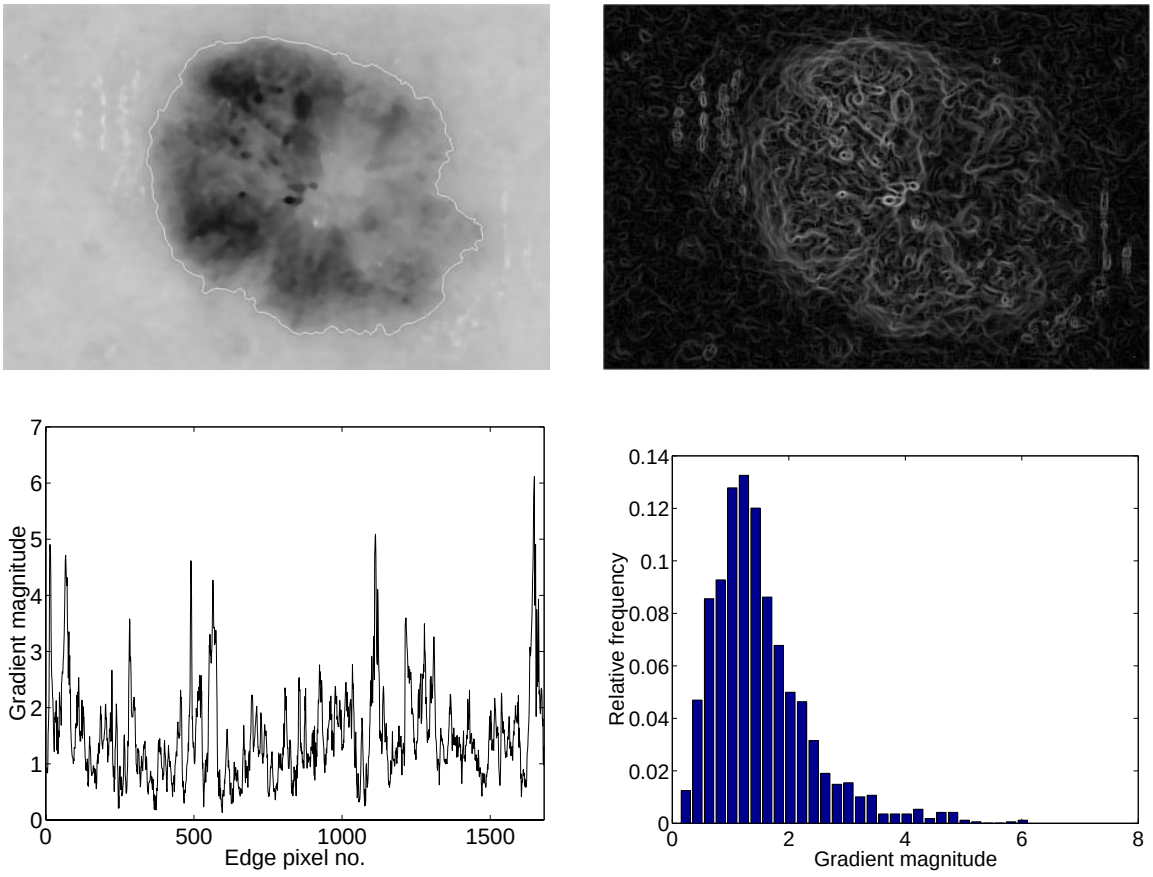

Figure 7: Example of measuring edge abruptness in a dermatoscopic image. Upper left: Intensity image showing the lesion edge obtained from the segmentation process. Upper right: Gradient magnitude image. Note: The gradient magnitude range has been compressed by the transformation, $g_c(m,n) = \log(1 + g(m,n))$, in order to enhance the visual quality. Lower left: The gradient magnitude sampled along the lesion edge. Lower right: Histogram of gradient magnitude measured along the lesion edge.

$$m_e = \frac{1}{K}\sum_{k=0}^{K-1} e(k), \quad v_e = \frac{1}{K}\sum_{k=0}^{K-1} e^2(k) - m_e^2, \tag{26}$$

where the sample mean, m_e, describes the general abruptness level and the sample variance, v_e, describes the variation of the abruptness along the skin lesion edge.

In figure 7, an example of measuring the abruptness in a dermatoscopic image is shown.

Skin lesion specific comments

As mentioned previously, several skin lesions larger than the field of view of the camera are included in this work. For these lesions the gradient magnitude has not been sampled along false edges. These occur at the boundaries of the image where the skin lesion crosses the image border, see the example in figure 6. Thus we assume, that enough edge information is available from the visible part of the skin lesion in

order to describe the characteristics of the lesion edge and that we can neglect the contributions outside the field of view.

2.4.3 Color

The color distribution of a skin lesion is another important aspect that may contribute to an accurate diagnosis. Dermatologists have identified 6 shades of color that may be present in skin lesions examined with the dermatoscopic imaging technique. These colors arise due to several biological processes [10]. The colors are: *Light-brown*, *dark-brown*, *white*, *red*, *blue* and *black* [10]. This is a rather vague color description that is likely to cause some discrepancies between how different individuals perceive skin lesion colors. There are especially problems with separating light-brown from dark-brown but problems also occur with red and dark-brown due to a rather reddish glow of the dark-brown color in skin lesions.

We will nevertheless try to define a consistent method of measuring skin lesion colors that matches dermatologists intuitive perception of colors. This is done by defining color prototypes that are in close correspondence with the color perception of dermatologists and using these prototypes to determine the color contents of skin lesions. As a guideline, a large number of colors is considered to be an indicator of malignancy.

Color prototype determination

The color prototypes have been determined from three 2-D histograms[7] of 18 randomly selected skin lesion images combined into one large image. By inspecting the histograms, several clusters matching the color perception of dermatologists have been defined and the perceived cluster centers are used as prototypes. This is shown in figure 8. Note, that several shades of light-brown, dark-brown and blue have been identified. No reliable prototype for red distinguishing it from dark-brown could be determined. This is a problem also found among dermatologists. One may consider a part of a lesion to be red while another may suggest dark-brown. Due to these difficulties, a red prototype has not been defined.

It is clear that this way of determining prototypes is a very subjective process, yet great care has been taken in order for the prototypes to match the color perception of dermatologists[8].

A standard *k-means* clustering algorithm using the Euclidean distance measure in the RGB color space has also been employed but did not yield acceptable color prototypes. It is obvious from inspecting the 2-D histograms that the Euclidean distance measure is not the most appropriate choice due to the varying shape of the different clusters. It would be beneficial to allow the distance measure to vary between clusters acknowledging that different probability distributions generate the individual clusters.

[7] Red-green, red-blue and green-blue 2-D histograms.

[8] The author has spent hour-long sessions with dermatologists viewing and discussing skin lesions in order to gain insight into their color perception.

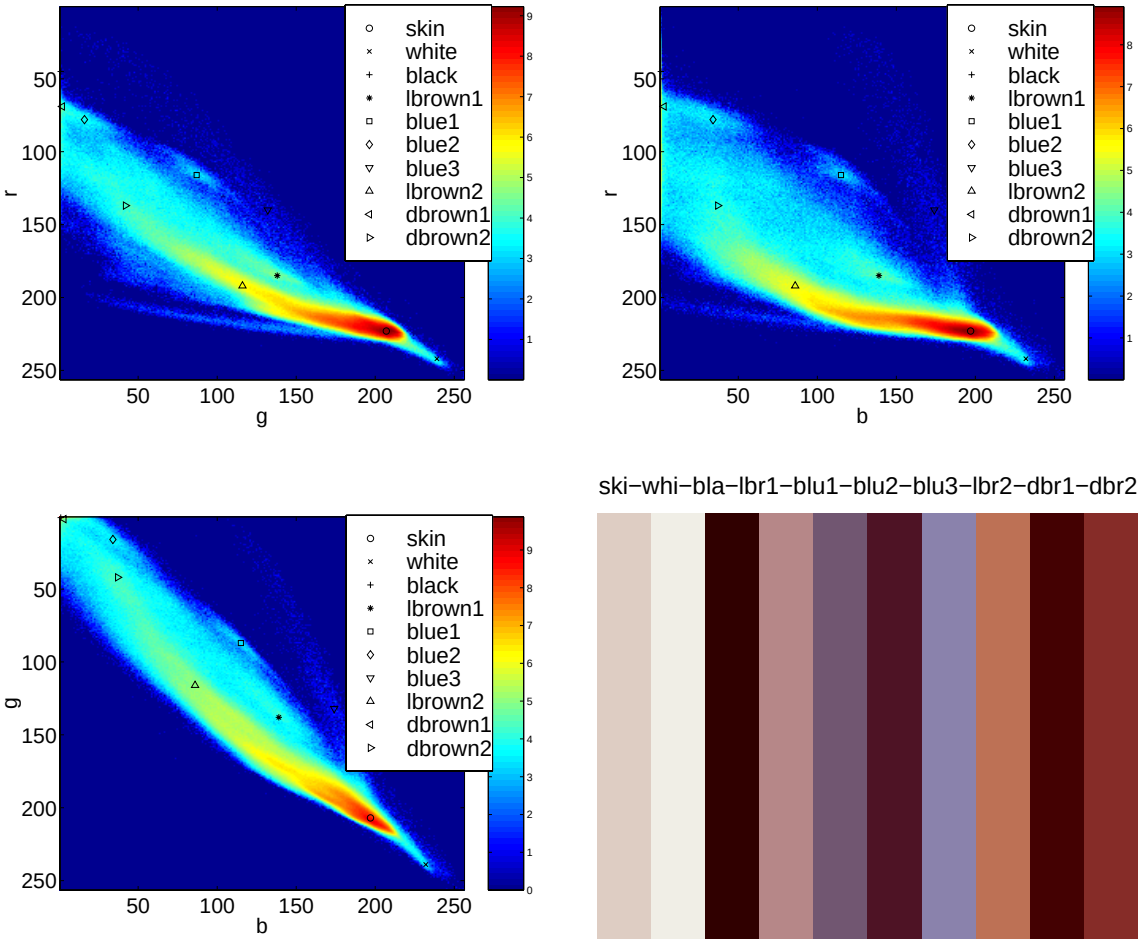

Figure 8: Color prototypes have been found manually by inspecting the combined 2-D histograms of 18 randomly selected images. The perceived cluster centers are chosen as prototypes. Upper left: Red-green 2-D histogram. The histogram values, $h(r, g)$, have been compressed by the transformation, $h_c(r, g) = \log(1 + h(r, g))$, in order to enhance the visual quality. Upper right: Red-blue 2-D histogram (log-transformed). Lower left: Green-blue 2-D histogram (log-transformed). Lower right: The determined color prototypes. The *skin* color prototype is left out since it is eliminated by the segmentation process. Only colors inside the lesion are of interest in this work.

Often these distributions may be considered Gaussian, see e.g. [19].

Another contributing factor to the failure of the standard *k-means* algorithm is the number of pixels in each cluster. The histograms in figure 8 are log-transformed, that is, the dynamic range has been compressed in order to enhance the visual quality. Thus the number of pixels close to the center of some of the clusters seems relative large compared to, e.g., the dominant *skin* color cluster even though the number of pixels in these clusters is in fact rather small. In the standard *k-means* algorithm these clusters are likely to be suppressed by the higher populated dominant clusters resulting in unacceptable results.

Thus in order to overcome these problems and to incorporate the color perception of dermatologists, the manually selected prototypes are used in this work. Note, that 10 color clusters have been defined but only 9 prototypes are used. The *skin* color prototype is left out as this color is eliminated by the segmentation process and normally only found outside the lesion. The 9 color prototypes thus corresponds of *white*, *black*, *light-brown 1*, *light-brown 2*, *dark-brown 1*, *dark-brown 2*, *blue 1*, *blue 2* and *blue 3* representing 5 different colors.

Measuring color

The color contents of a skin lesion may be determined by comparing the skin lesion pixels with color prototypes. Here we will use the Euclidean distance measure for comparing colors,

$$d_i^2(m,n) = [r(m,n) - r_i]^2 + [g(m,n) - g_i]^2 + [b(m,n) - b_i]^2, \quad i = 1, 2, \ldots 9, \tag{27}$$

where $d_i(m,n)$ is the distance in RGB colorspace from pixel (m,n) to the $i'th$ color prototype defined by $cp_i = [r_i \; g_i \; b_i]^{\mathrm{T}}$.

Every skin lesion pixel can now be assigned a prototype color by selecting the shortest distance. That is, the pixel (m,n) should be assigned the prototype color cp_i if

$$d_i(m,n) < d_j(m,n) \quad \text{for all } i \neq j. \tag{28}$$

We may now describe the color contents of a skin lesion as a set of relative areas - one for each color prototype. This may be written as

$$a_i = \frac{A_{cp_i}}{A}, \tag{29}$$

where A is the area of the skin lesion, A_{cp_i} the area inside the skin lesion occupied by pixels close to prototype color cp_i as defined by equation (28) and a_i the relative measure of the color content of the prototype color cp_i. Since we do not wish to distinguish between different shades of the same color, the

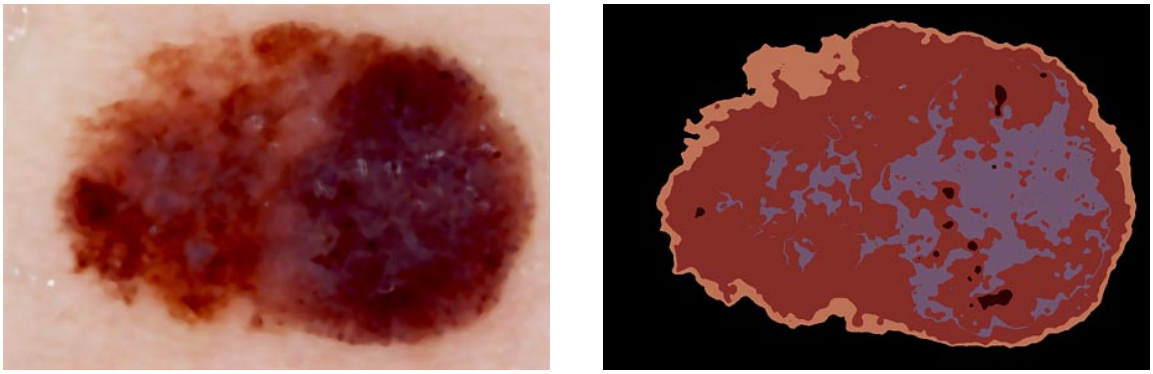

Figure 9: Examples of color detection in a dermatoscopic image. Left: Original median filtered image. Right: Results of comparing the skin lesion image in the left panel with color prototypes in the RGB colorspace using the Euclidean difference measure. Note, that all shades of blue are representated by the *blue1* prototype seen in figure 8, all shades of dark-brown by *dbrown2* and all shades of light-brown by *lbrown2*.

color content of light-brown is defined as the sum of a_i for the two light-brown color shades. The same applies to the blue and dark-brown color shades.

As mentioned in the previous section, the choice of distance measure is not trivial. The most appropriate distance measure in this context would be one that takes the color perception of dermatologists into account. The CIE[9] has proposed the perceptually uniform colorspaces, CIE-Lab and CIE-Luv, in which the Euclidean distance measure matches the average humans perception of color differences [20]. In order to transform pixels in RGB colorspace to either CIE-Luv or CIE-Lab colorspace, one must first empirically determine a linear 3×3 transformation matrix for the complete imaging system[10] that transforms the RGB colorspace of the imaging system to the standardized CIE-RGB colorspace, see e.g. [21]. The CIE-RGB values may then be converted through a non-linear transformation into either CIE-Luv or CIE-Lab values [17]. Using the Euclidean distance measure in either of these colorspaces for comparing colors may yield results corresponding better with the color perception of dermatologists.

An example of skin lesion comparison with the color prototypes is shown in figure 9.

Skin lesion specific comments

Note, that the use of color prototypes requires that the conditions of the imaging system are very controlled in order to achieve color consistency. This involves camera, lighting conditions, film type, film development process and scanner.

[9] Commission Internationale de L'Eclairage - the international committee on color standards.

[10] The imaging system in this application consists of camera, film, development process and image scanning.

3 A PROBABILISTIC FRAMEWORK FOR CLASSIFICATION

3.1 Bayes decision theory

Bayes decision theory is based on the assumption that the classification problem at hand can be expressed in probabilistic terms and that these terms are either known or can be estimated.

Suppose the classification problem is to map an input pattern $\mathbf{x}$ into a class $\mathcal{C}_l$ out of $n_\mathcal{C}$ classes where $l = 1, 2, \ldots, n_\mathcal{C}$. We can now define several probabilistic terms that are related through *Bayes' theorem* [22],

$$P(\mathcal{C}_l|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_l)P(\mathcal{C}_l)}{p(\mathbf{x})}. \tag{30}$$

$P(\mathcal{C}_l)$ is the class *prior* and reflects our prior belief of an unobserved pattern $\mathbf{x}$ belonging to class $\mathcal{C}_l$. $p(\mathbf{x}|\mathcal{C}_l)$ is the *class-conditional probability density function* and describes the probability characteristics of $\mathbf{x}$ once we know it belongs to class $\mathcal{C}_l$. The *posterior* probability is denoted by $P(\mathcal{C}_l|\mathbf{x})$ and is the probability of an observed pattern $\mathbf{x}$ belonging to class $\mathcal{C}_l$. The unconditional probability density function, $p(\mathbf{x})$, describing the density function for $\mathbf{x}$ regardless of the class, is given by

$$p(\mathbf{x}) = \sum_{l=1}^{n_\mathcal{C}} p(\mathbf{x}|\mathcal{C}_l)P(\mathcal{C}_l). \tag{31}$$

In short, Bayes' theorem shows how the observation of a pattern $\mathbf{x}$ changes the prior probability $P(\mathcal{C}_l)$ into a posterior probability $P(\mathcal{C}_l|\mathbf{x})$.

A classification system usually divides the input space into a set of $n_\mathcal{C}$ decision regions, $\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_{n_\mathcal{C}}$, so that a pattern, $\mathbf{x}$, located in $\mathcal{R}_l$ is assigned to class $\mathcal{C}_l$. The boundaries between the regions are called *decision boundaries*. Often the aim of a classifier is to minimize the probability of error, that is, to minimize the probability of classifying a pattern $\mathbf{x}$ belonging to class $\mathcal{C}_l$ as a different class due to $\mathbf{x}$ not being in decision region $\mathcal{R}_l$. This leads to *Bayes' minimum-error decision rule* saying that a pattern should be assigned to class $\mathcal{C}_l$ if [22]

$$P(\mathcal{C}_l|\mathbf{x}) > P(\mathcal{C}_m|\mathbf{x}) \qquad \text{for all } l \neq m. \tag{32}$$

As already mentioned, Bayes' minimum-error decision rule assumes that the aim is to minimize the probability of error. This makes sense if every possible error is associated with the same cost. If this is not the case, one could adopt a *risk-based approach*, see, e.g., [23]. It may also be appropriate not to divide the entire input space into $n_\mathcal{C}$ decision regions. If a pattern has a low posterior probability for all classes, it may be beneficial to reject the pattern, rather than assigning it to a class. This is called *error-reject trade-off*, see, e.g., [22], [24], [25].

3.2 Measuring model performance

Up until now, we have assumed that we either know the true posterior probabilities for the classes or that we have some estimate of the posterior probabilities. We will now introduce the notion of a model producing estimates of the posterior probabilities.

Assume we have a data set, $\mathcal{D}$, which we shall call a *training set*, consisting of $q_{\mathcal{D}}$ input-output pairs drawn from the joint probability distribution $p(\mathbf{x}, \mathbf{y})$

$$\mathcal{D} = \{(\mathbf{x}^{\mu}, \mathbf{y}^{\mu}) | \mu = 1, 2, \ldots, q_{\mathcal{D}}\}, \tag{33}$$

where $\mathbf{x}$ is an input pattern vector and $\mathbf{y}$ is an output vector containing the corresponding class label: $\mathbf{y}^{\mathrm{T}} = (y_1, y_2, \ldots, y_{n_{\mathcal{C}}})$ with $y_l = 1$, if $\mathbf{x} \in \mathcal{C}_l$, otherwise $y_l = 0$. This class labeling scheme is known as *1-of-$n_{\mathcal{C}}$* coding.

Let us also assume, we have a model, $\mathcal{M}$, parameterized by a vector, $\mathbf{u}$, that is estimated on the basis of the training set, $\mathcal{D}$, and let the model be capable of producing estimates of the posterior probabilities for the classes,

$$\mathcal{M}(\mathbf{u}) : \mathbf{x} \curvearrowright \hat{\mathbf{y}}, \tag{34}$$

where $\hat{\mathbf{y}}^{\mathrm{T}} = (\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_{n_{\mathcal{C}}})$ contains estimates of the true posterior probabilities, i.e., $\hat{y}_l = \hat{P}(\mathcal{C}_l|\mathbf{x})$.

We can now use Bayes' theorem to define several probabilistic terms for the model $\mathcal{M}$,

$$p(\mathbf{u}|\mathcal{D}) = \frac{p(\mathcal{D}|\mathbf{u})p(\mathbf{u})}{p(\mathcal{D})}. \tag{35}$$

$p(\mathbf{u})$ is the *parameter prior* and reflects our prior knowledge of the model parameters before observing any data. $p(\mathcal{D}|\mathbf{u})$ is the *likelihood of the model* and describes how probable it is that the data, $\mathcal{D}$, is generated by the model parameterized by $\mathbf{u}$. The *posterior parameter distribution* is denoted by $p(\mathbf{u}|\mathcal{D})$ and quantifies the probability distribution of the model parameters once the data has been observed. The unconditional probability distribution, $p(\mathcal{D})$, is a normalization factor given by $p(\mathcal{D}) = \int p(\mathcal{D}|\mathbf{u})p(\mathbf{u})d\mathbf{u}$.

Now, in order to design a model as close to the true underlying model as possible, we may find the parameters that maximize the posterior parameter distribution,

$$\hat{\mathbf{u}}^{\mathrm{MAP}} = \arg\max_{\mathbf{u}} \left[p(\mathcal{D}|\mathbf{u})p(\mathbf{u})\right]. \tag{36}$$

This is known as *maximum a posteriori* (MAP) estimation.

If we have a uniform parameter prior, $p(\mathbf{u})$, the MAP estimate reduces to the *maximum likelihood* (ML) estimate,

$$\hat{\mathbf{u}}^{\mathrm{ML}} = \arg\max_{\mathbf{u}} \left[p(\mathcal{D}|\mathbf{u}) \right]. \tag{37}$$

The MAP and ML estimate is based on the assumption that there is one near-optimal model matching the true model the best. Bayesians argue that one should use the entire posterior parameter distribution as a description of the model when doing output predictions. Examples of Bayesian approaches include David MacKay's *Bayesian framework for classification* based on approximating the posterior weight distribution [26], [27], [28] and *Markov Chain Monte Carlo* schemes based on sampling the posterior weight distribution [29], [30]. We will pursue the ML principle.

Assuming that the individual samples in $\mathcal{D}$ are drawn independently, the likelihood of the model can be written as

$$p(\mathcal{D}|\mathbf{u}) = \prod_{\mu=1}^{q_{\mathcal{D}}} p(\mathbf{y}^{\mu}|\mathbf{x}^{\mu}, \mathbf{u}) p(\mathbf{x}^{\mu}). \tag{38}$$

Instead of maximizing the likelihood, we may choose to minimize the negative logarithm[11] of the likelihood

$$-\log p(\mathcal{D}|\mathbf{u}) \quad = \quad -\sum_{\mu=1}^{q_{\mathcal{D}}} \left[\log p(\mathbf{y}^{\mu}|\mathbf{x}^{\mu}, \mathbf{u}) + \log p(\mathbf{x}^{\mu}) \right]. \tag{39}$$

Since $p(\mathbf{x})$ is independent of the parameter vector, $\mathbf{u}$, we can discard this term from equation (39) and minimize the following function instead,

$$E_{\mathcal{D}}(\mathbf{u}) \quad = \quad -\frac{1}{q_{\mathcal{D}}} \sum_{\mu=1}^{q_{\mathcal{D}}} \log p(\mathbf{y}^{\mu}|\mathbf{x}^{\mu}, \mathbf{u}) \tag{40}$$

$$= \quad \frac{1}{q_{\mathcal{D}}} \sum_{\mu=1}^{q_{\mathcal{D}}} e(\mathbf{x}^{\mu}, \mathbf{y}^{\mu}, \mathbf{u}), \tag{41}$$

where $E_{\mathcal{D}}(\mathbf{u})$ is called an *error function* and $e(\mathbf{x}, \mathbf{y}, \mathbf{u})$ a *loss function*. Note, that the negative log-likelihood has been normalized with the number of samples in the training set $\mathcal{D}$, thus making $E_{\mathcal{D}}(\mathbf{u})$ an expression of the average pattern error.

Now, let us return to the MAP technique. As with the ML estimate, instead of maximizing the posterior parameter distribution, we can choose to minimize the negative logarithm of the posterior parameter distribution

[11] Since the logarithm is a monotonic function, the two approaches lead to the same results.

$$-\log p(\mathcal{D}|\mathbf{u}) - \log p(\mathbf{u}) = -\sum_{\mu=1}^{q_\mathcal{D}} [\log p(\mathbf{y}^\mu|\mathbf{x}^\mu, \mathbf{u}) + \log p(\mathbf{x}^\mu)] - \log p(\mathbf{u}). \tag{42}$$

Again we note that $p(\mathbf{x})$ is independent of $\mathbf{u}$, so we may discard this term and minimize the following function instead,

$$-\frac{1}{q_\mathcal{D}} \sum_{\mu=1}^{q_\mathcal{D}} \log p(\mathbf{y}^\mu|\mathbf{x}^\mu, \mathbf{u}) - \frac{1}{q_\mathcal{D}} \log p(\mathbf{u}). \tag{43}$$

This function that we wish to minimize can now be written as

$$C(\mathbf{u}) = E_\mathcal{D}(\mathbf{u}) + R(\mathbf{u}), \tag{44}$$

where $C(\mathbf{u})$ is called a *cost function* and $R(\mathbf{u}) \propto -\frac{1}{q_\mathcal{D}} \log p(\mathbf{u})$ a *regularization function*. The latter is determined by the parameter prior and we shall return to this subject in section 3.4.1.

In the next section, we will derive a loss function for multiple-class problems based on the ML principle.

3.2.1 Cross-entropy error function for multiple classes

We will now consider the case where we have multiple exclusive classes, i.e., a pattern belongs to one and only one class. As in section 3.2, we assume that we have a model capable of producing estimates of the true posterior probabilities for the classes: $\hat{y}_l = \hat{P}(\mathcal{C}_l|\mathbf{x})$, we use a 1-of-$n_\mathcal{C}$ coding scheme for the class labeling and the distributions of the different class labels, y_l, are independent. The probability of observing a class label, $\mathbf{y}$, given a pattern, $\mathbf{x}$, is $\hat{P}(\mathcal{C}_l|\mathbf{x})$, if the true class is $\mathcal{C}_l$, which can be written as

$$p(\mathbf{y}|\mathbf{x}, \mathbf{u}) = \prod_{l=1}^{n_\mathcal{C}} (\hat{y}_l)^{y_l}. \tag{45}$$

Inserting equation (45) in equation (40), we obtain the following error function,

$$E_\mathcal{D}(\mathbf{u}) \quad = \quad -\frac{1}{q_\mathcal{D}} \sum_{\mu=1}^{q_\mathcal{D}} \sum_{l=1}^{n_\mathcal{C}} y_l^\mu \log \hat{y}_l^\mu, \tag{46}$$

which is known as the *cross-entropy* error function [23].

3.3 Measuring generalization performance

When modeling, we would like our model to be as close as possible to the true model described by $p(\mathbf{x}, \mathbf{y})$. In order to measure this, we define the *generalization ability* of a model as its ability to predict the output of the true model. Thus, the *generalization error* of a model can be defined as

$$G(\mathbf{u}) = \langle e(\mathbf{x}, \mathbf{y}, \mathbf{u}) \rangle_{p(\mathbf{x},\mathbf{y})} \tag{47}$$

$$= \int e(\mathbf{x}, \mathbf{y}, \mathbf{u}) p(\mathbf{x}, \mathbf{y}) d\mathbf{x} d\mathbf{y} \tag{48}$$

where the loss function, $e(\mathbf{x}, \mathbf{y}, \mathbf{u})$, could be, e.g, the cross-entropy error. The lower bound of $G(\mathbf{u})$ is $G(\mathbf{u}^\star)$, where $\mathbf{u}^\star$ denotes the parameters of the true model.

In the limit of an infinite training set, $\mathcal{D}$, the training error converges to the generalization error,

$$\lim_{q_\mathcal{D} \to \infty} E_\mathcal{D}(\mathbf{u}) = \lim_{q_\mathcal{D} \to \infty} \frac{1}{q_\mathcal{D}} \sum_{\mu=1}^{q_\mathcal{D}} e(\mathbf{x}^\mu, \mathbf{y}^\mu, \mathbf{u}) \tag{49}$$

$$= \int e(\mathbf{x}, \mathbf{y}, \mathbf{u}) p(\mathbf{x}, \mathbf{y}) d\mathbf{x} d\mathbf{y}. \tag{50}$$

Note that $G(\mathbf{u})$ is dependent on the training set through the model parameters $\mathbf{u}$. We may remove this dependency by defining the *expected generalization error* as the average generalization error w.r.t. all possible training sets of size $q_\mathcal{D}$,

$$\bar{G} = \langle G(\mathbf{u}) \rangle_{p(\mathcal{D})} \tag{51}$$

$$= \int G(\mathbf{u}) p(\mathcal{D}) d\mathcal{D}. \tag{52}$$

Here we have acknowledged that the generalization error itself is a stochastic variable and defined the expected or average generalization error. We could equally well have defined other interesting measures like, e.g., the median. See [31] for a discussion of different generalization error statistics.

Usually, we do not know the true joint input-output distribution, $p(\mathbf{x}, \mathbf{y})$, and thus cannot determine neither $G(\mathbf{u})$ or $\bar{G}$. Instead, we can compute either empirical or algebraic estimates of these quantities which we shall discuss in the next two sections.

3.3.1 Empirical estimates

Sine we usually cannot assess the true joint input-output distribution, $p(\mathbf{x}, \mathbf{y})$, we may resolve to using empirical estimates of this distribution.

One such estimator is obtained by employing a data set that is independent of the training set but drawn from the same true distribution $p(\mathbf{x}, \mathbf{y})$. We call this a *test set*,

$$\mathcal{T} = \{(\mathbf{x}^\mu, \mathbf{y}^\mu | \mu = 1, 2, \ldots, q_\mathcal{T}). \tag{53}$$

If we use the empirical joint input-output distribution, $\hat{p}_\mathcal{T}(\mathbf{x}, \mathbf{y})$, based on the test set, we may now use the *test error* as an estimate of the generalization error,

$$\hat{G}_{\mathcal{T}}(\mathbf{u}) = \frac{1}{q_{\mathcal{T}}} \sum_{\mu=1}^{q_{\mathcal{T}}} e(\mathbf{x}^{\mu}, \mathbf{y}^{\mu}, \mathbf{u}). \tag{54}$$

As with the training error, $\hat{G}_{\mathcal{T}}(\mathbf{u})$ converges to the generalization error, $G(\mathbf{u})$, when the test set, $\mathcal{T}$, is infinite.

Now, we would like the training set to be as large as possible in order to create an accurate model. At the same time the test set should be large in order to get a reliable estimate of the generalization ability of the model. Unfortunately, the available data is usually rather limited, so we have to deal with a trade-off between having a large training set and a large test set. A method trying to overcome this trade-off is called *cross-validation* [32], [33]. The idea of cross-validation is based on training and testing on disjunct subsets of data resampled from the available database. If we split the database up into K disjunct data sets, we may estimate a model using $K-1$ sets and evaluate its performance on the remaining set. This can be done K times resulting in K different models with K measures of the generalization performance. The cross-validation error is then defined as

$$\hat{G}_{\mathrm{CV}} = \frac{1}{K} \sum_{i=1}^{K} \hat{G}_{\mathcal{T}^{(i)}}(\mathbf{u}^{(i)}) \tag{55}$$

where $\hat{G}_{\mathcal{T}^{(i)}}(\mathbf{u}^{(i)})$ is the test error defined by equation(54) and i the split label. This provides us with an estimate of the expected generalization error defined by equation(51).

If each of the K disjunct data sets only contains one pattern, we obtain the special case called *leave-one-out cross-validation*.

Cross-validation has one major drawback, though, and that is the high computational costs involved. K models have to be estimated which for leave-one-out cross-validation corresponds to estimating as many models as there are available patterns in the data set. A scheme trying to remedy this based on linear unlearning of patterns has been proposed in [34]. An application using this technique is presented in [35].

3.3.2 Algebraic estimates

Empirical generalization error estimates require a fraction of the available data to be set aside thus reducing the amount of data available for the training set. And as stated previously, we would prefer a large training set in order to model the true model as accurately as possible.

In order to maximize the size of the training set, we will now consider an algebraic estimate of the average generalization error based only on the data in the training set. We will assume the following:

- Independence of input and error on output.

- There exists a set of parameters, $\mathbf{u}^\star$, that implements the true model, i.e., the chosen model architecture should be capable of implementing the true model.
- The number of patterns in the training set is large.

Under these assumptions, the following estimate of the average generalization error can be derived [36], [37], [38],

$$\langle G(\mathbf{u}) \rangle_{p(\mathcal{D})} \approx E_{\mathcal{D}}(\mathbf{u}) + \frac{1}{q_{\mathcal{D}}} \mathrm{tr}\left[\mathbf{J}^{-1}\mathbf{H}\right], \tag{56}$$

where $\mathbf{H}$ and $\mathbf{J}$ is the Hessian matrix for the unregularized and regularized cost function, respectively.

This estimate may be used to select an optimal model among a hierarchy of models with decreasing complexity, i.e., every model should be a sub model of the previous model in the hierarchy [38].

For other texts on algebraic generalization error estimates, see, e.g., [39], [40], [41], [42].

3.4 Controlling model complexity

When estimating models, we face the problem of choosing a model that has an appropriate complexity. That is, the model should be flexible enough to adequately model the underlying function of the true model. At the same time, we should ensure that the model is not too flexible in order not to capture the noise in the data. The latter case is known as *overfitting* [43], [23].

In brief, the purpose with controlling the model complexity is to maximize the generalization performance of the model. We will in the next two sections consider two such techniques based on parameter regularization and parameter pruning, respectively. Both methods are based on the assumption that the model is too complex.

3.4.1 Weight decay regularization

As we saw in section 3.2, the MAP technique involves a prior for the model parameters and the cost function could thus be written as

$$C(\mathbf{u}) = E_{\mathcal{D}}(\mathbf{u}) + R(\mathbf{u}) \tag{57}$$

where $R(\mathbf{u}) \propto -\frac{1}{q_{\mathcal{D}}} \log p(\mathbf{u})$ is called a *regularization function*.

In order to avoid overfitting, we should consider a prior that has the potential of limiting the model complexity by ensuring that the decision boundaries are smooth. One such prior that favors small parameters[12] is a zero mean Gaussian parameter prior with the individual parameters being independent,

[12] Here we assume that small parameters lead to very constrained models while large parameters allow very flexible models which will be the case for the neural network models considered in section 4.

$$p(u_k) = \frac{1}{\sqrt{(2\pi)1/\alpha_k}} \exp\left[-\frac{1}{2}\alpha_k u_k^2\right], \tag{58}$$

where α_k is the inverse prior parameter variance that can be used for controlling the range of u_k. We can now write the normalized negative logarithm of the parameter prior as

$$-\frac{1}{q_{\mathcal{D}}} \log p(\mathbf{u}) = -\frac{1}{q_{\mathcal{D}}} \sum_{k=1}^{n_u} \log p(u_k) \tag{59}$$

$$= -\frac{1}{q_{\mathcal{D}}} \sum_{k=1}^{n_u} \left[-\frac{1}{2}\alpha_k u_k^2 - \log\frac{2\pi}{\alpha_k}\right], \tag{60}$$

where n_u is the total number of parameters.

We have seen from the MAP estimate that $R(\mathbf{u})$ should really equal $-\frac{1}{q_{\mathcal{D}}} \log p(\mathbf{u})$, but since the second term in equation (60), $\log\frac{2\pi}{\alpha_k}$, doesn't depend on $\mathbf{u}$ and we want to minimize $C(\mathbf{u}) = E_{\mathcal{D}}(\mathbf{u}) + R(\mathbf{u})$ with respect to $\mathbf{u}$, we may discard this term and define the regularization function as

$$R(\mathbf{u}) = \frac{1}{2q_{\mathcal{D}}} \sum_{k=1}^{n_u} \alpha_k u_k^2 = \frac{1}{2}\mathbf{u}^{\mathrm{T}}\mathbf{R}\mathbf{u}, \tag{61}$$

where $\mathbf{R}$ is a diagonal positive semidefinite matrix with elements $\alpha_k/q_{\mathcal{D}}$ in the diagonal. This particular form of the regularization function is called *weight decay* in the neural network community since it penalizes large parameters or weights whereas it for regression problems in traditional statistics is known as *ridge regression* when all α_k's are equal [44]. The regularization parameters, α_k, are also known as *hyperparameters* since they themselves control other parameters, in this case, the model parameters.

3.4.2 Optimal brain damage pruning

As we saw previously, we could limit the effect of a parameter or implicitly remove it by setting its regularization parameter, i.e., the inverse parameter variance, to a very large value. We could instead explicitly remove a parameter using one of several pruning techniques.

These methods are often based on computing the importance of each parameter by estimating the increase in an error measure that the removal of a parameter causes. All parameters are then ranked according to their importance denoted *saliency* and a percentage of the parameters with the lowest saliencies can be removed. The model is then re-estimated and the procedure is repeated until no parameters remain. This results in a family of models with decreasing complexity. For each model, an estimate of the generalization error may be computed and used for selecting the optimal model.

We will consider a pruning technique called *optimal brain damage* [45], that is based on the following assumptions:

- The regularized cost function is at a minimum.
- The terms of third and higher order in a Taylor expansion of the error and regularized cost function can be neglected.
- The off-diagonal elements in the Hessian matrix can be neglected if more than one parameter is removed.

Under these assumptions the OBD saliency for a weight, $\hat{u}_k$, is

$$s_k^{\text{OBD}} = \left(\frac{\alpha_k}{q_{\mathcal{D}}} + \frac{1}{2} \mathbf{H}_{kk} \right) \hat{u}_k^2, \tag{62}$$

where $\mathbf{H}_{kk}$ is the $k'th$ diagonal element of the Hessian matrix.

4 NEURAL CLASSIFIER MODELING

The traditional approach to classification is statistical and concerns the modeling of stationary class-conditional probability distributions by a set of basis functions, e.g., Parzen windows or Gaussian mixtures [22], [23], [18].

Neural networks have in the last decade been employed extensively for classification applications. The two most common neural network architectures for supervised classification are the multi-layer perceptron and the radial basis function network with two layers of weights. We will consider the multi-layer perceptron architecture in greater detail in the next section.

Both classes of neural networks possess the important *universal approximation* capability, i.e., they may approximate any given function[13] with arbitrary precision as long as the number of hidden units are large enough [46],[18]. Since neural networks *learn by example*, they are particular effective in situations where no suitable traditional statistical model may be identified, i.e., knowledge about the true data-generating system is poor.

Radial basis function networks will not be discussed any further. For a more thorough introduction, see, e.g., [23].

4.1 Multi-layer perceptron architecture

We will now focus on two-layer perceptrons and define a particular model architecture that is used throughout the rest of this work.

[13] If the network output function imposes bounds on the the output values, the networks can of course only approximate equally bounded functions.

The hidden unit activation function used is the hyperbolic tangent function. Thus, the output of the hidden units for a pattern, $\mathbf{x}^\mu$, may be written as

$$h_j(\mathbf{x}^\mu) = \tanh\left(\sum_{k=1}^{n_I} w_{jk}^I x_k^\mu + w_{j0}^I\right), \quad j = 1, 2, \ldots, n_H, \tag{63}$$

where w_{jk}^I is the weight connecting input k and hidden unit j, w_{j0}^I is the threshold for hidden unit j, n_I is the number of inputs and n_H is the number of hidden units.

The hidden unit outputs are weighted and summed, yielding the following unbounded network outputs,

$$\phi_i(\mathbf{x}^\mu) = \sum_{j=1}^{n_H} w_{ij}^H h_j(\mathbf{x}^\mu) + w_{i0}^H, \quad i = 1, 2, \ldots, n_O, \tag{64}$$

where w_{ij}^H is the weight connecting hidden unit j and the unbounded output unit i, w_{i0}^H is the threshold for the unbounded output unit i and n_O is the number of unbounded output units.

In order to employ the probabilistic framework derived in section 3, the neural classifier outputs must be normalized so that the classifier may be used for estimating posterior probabilities. We will now consider two slightly different normalization schemes and discuss their properties.

4.1.1 Softmax normalization

The standard way of ensuring that network outputs may be interpreted as probabilities is by using the normalized exponential transformation know as *softmax* [47],

$$\hat{y}_i^\mu = \hat{P}(\mathcal{C}_i|\mathbf{x}^\mu) = \frac{\exp[\phi_i(\mathbf{x}^\mu)]}{\sum_{i'=1}^{n_O} \exp[\phi_{i'}(\mathbf{x}^\mu)]}, \tag{65}$$

where $\hat{y}_i^\mu$ is short for the estimated posterior probability that the pattern, $\mathbf{x}^\mu$, belongs to class $\mathcal{C}_i$. We thus have the following properties: $0 \leq \hat{y}_i^\mu \leq 1$, $\sum_{i=1}^{n_O} \hat{y}_i^\mu = 1$. As can be seen, the softmax normalization introduces a redundancy in the output representation due to the property that the posterior probability estimates for a pattern sum to one.

An effect of this is that the unregularized Hessian matrix for a well-trained network will be singular due to the output redundancy resulting in a dependency between the weights going to one output unit and the weights going to the other output units. This effectively reduces the rank of the unregularized Hessian matrix by the number of hidden units plus one (threshold unit). Any computations involving the inverse Hessian matrix will be affected by this. The problem is reduced by employing regularization since this usually reestablishes the full rank of the regularized Hessian.

The standard softmax network is shown in figure 10.

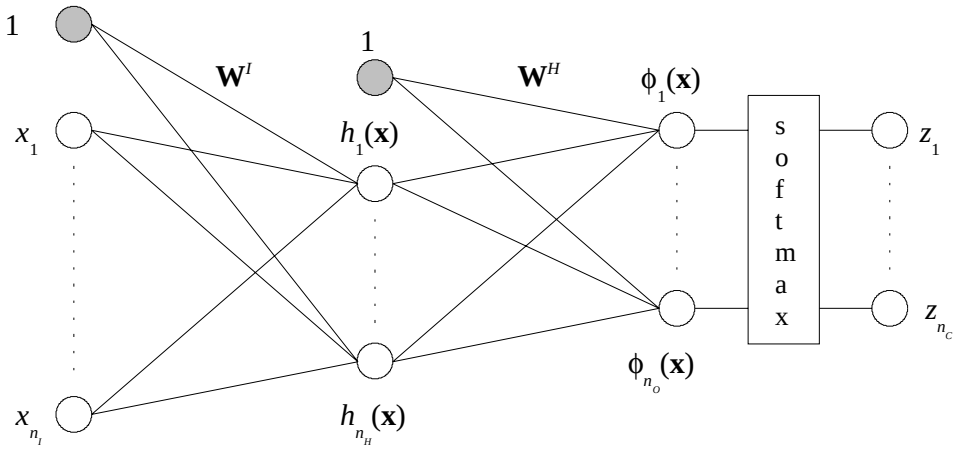

Figure 10: The standard two-layer softmax network. This has an inherent output redundancy yielding the unregularized Hessian singular for a well-trained network.

4.1.2 Modified softmax normalization

In order to remove the output redundancy introduced by the standard softmax normalization, we may remove one of the unbounded outputs. This yields the following *modified softmax* normalization,

$$\hat{y}_i^\mu = \begin{cases} \frac{\exp[\phi_i(\mathbf{x}^\mu)]}{1+\sum_{i'=1}^{n_\mathcal{C}-1}\exp[\phi_{i'}(\mathbf{x}^\mu)]}, & \text{for } i = 1, 2, \ldots, n_\mathcal{C}-1 \\ 1 - \sum_{i'=1}^{n_\mathcal{C}-1} \hat{y}_i^\mu, & \text{for } i = n_\mathcal{C} \end{cases}, \tag{66}$$

where $n_\mathcal{C}$ is the number of classes.

Another way of obtaining this modification is by removing all input connections to the unbounded output $\phi_{n_\mathcal{C}}(\cdot)$ and setting $\phi_{n_\mathcal{C}}(\cdot)$ to zero for all input patterns. Using the standard softmax normalization (65), we now effectively obtain the modified softmax normalization. This is illustrated in figure 11.

The modified softmax normalization has several benefits compared to the standard softmax normalization. With a certain number of hidden units, the modified softmax normalization reduces the network complexity, i.e., the number of weight parameters is reduced by the number of hidden units plus one. This improves the number of training patterns per weight relationship. The dependency between weights is removed, thus improving the performance of algorithms based on the computation of the inverse Hessian, e.g., the Newton scheme of updating weights that will be discussed in section 4.2.2. Another example where the modified softmax normalization may be beneficial is in MacKay's Bayesian framework for classification [26], [27]. This framework approximates the posterior probability distribution of the weights by a Gaussian distribution centered on the MAP solution of the weights and with the inverse Hessian as covariance matrix. The posterior class probabilities are then found by using the entire posterior weight distribution. Any inaccuracies in the Hessian may in this framework affect the results considerably.

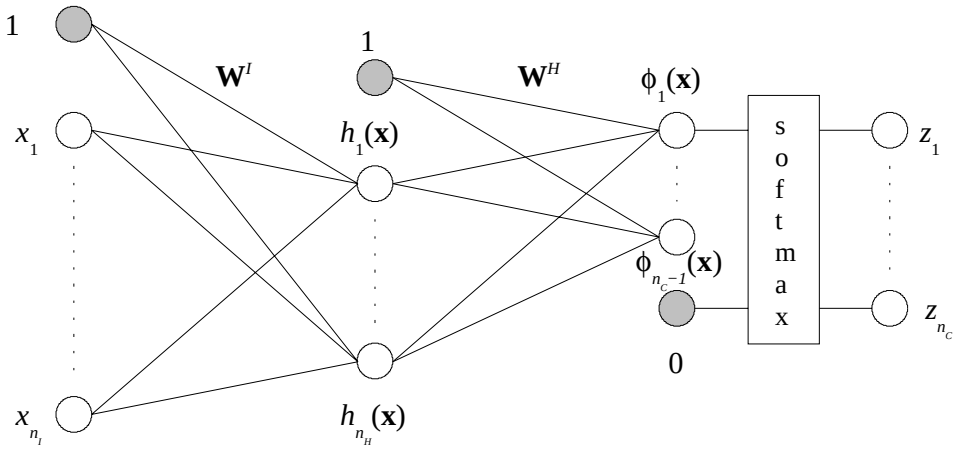

Figure 11: The modified two-layer softmax network. This does not have the inherent output redundancy.

The modified softmax scheme is recommended for output normalization.

4.2 Estimating model parameters

With the neural classifier architecture in place, we now need to address the task of estimating the model parameters. We will pursue the MAP approach with a Gaussian weight prior. As we recall from section 3, this yields the following cost function

$$C(\mathbf{u}) = E_{\mathcal{D}}(\mathbf{u}) + R(\mathbf{u}), \tag{67}$$

where $\mathbf{u}$ is a column vector containing all n_u network weights and thresholds, $E_{\mathcal{D}}(\mathbf{u})$ the cross-entropy error function (46) and $R(\mathbf{u})$ a regularization function proportional to the log weight prior.

The MAP solution for the network weights and thresholds will be found by using a training set, $\mathcal{D}$, of size $q_{\mathcal{D}}$ and by using optimization methods based on gradient and curvature information.

Common for these approaches is an iterative weight updating scheme that may be formulated as

$$\mathbf{u}^{(t+1)} = \mathbf{u}^{(t)} + \Delta\mathbf{u}^{(t)}, \tag{68}$$

where t indicates the iteration timestep and $\Delta\mathbf{u}^{(t)}$ the weight parameter change.

In the following sections, we will need the first and second derivatives of the cross-entropy error function w.r.t. the weights. Note, if the modified softmax normalization is used then $\phi_{n_C}(\mathbf{x}^\mu) = 0$ in the following.

The gradient is

$$\frac{\partial E_{\mathcal{D}}(\mathbf{u})}{\partial u_j} = -\frac{1}{q_{\mathcal{D}}} \sum_{\mu=1}^{q_{\mathcal{D}}} \sum_{i=1}^{n_{\mathcal{C}}} \frac{y_i^\mu}{\hat{y}_i^\mu} \frac{\partial \hat{y}_i^\mu}{\partial u_j}, \tag{69}$$

and the Hessian,

$$\frac{\partial^2 E_{\mathcal{D}}(\mathbf{u})}{\partial u_j \partial u_k} = -\frac{1}{q_{\mathcal{D}}} \sum_{\mu=1}^{q_{\mathcal{D}}} \sum_{i=1}^{n_{\mathcal{C}}} \frac{y_i^\mu}{\hat{y}_i^\mu} \left[-\frac{1}{\hat{y}_i^\mu} \frac{\partial \hat{y}_i^\mu}{\partial u_k} \frac{\partial \hat{y}_i^\mu}{\partial u_j} + \frac{\partial^2 \hat{y}_i^\mu}{\partial u_j \partial u_k} \right], \tag{70}$$

where the derivative of the posterior probability w.r.t. the weights is given by

$$\frac{\partial \hat{y}_i^\mu}{\partial u_j} = \hat{y}_i^\mu \sum_{i'=1}^{n_{\mathcal{C}}} (\delta_{i,i'} - \hat{y}_{i'}^\mu) \frac{\partial \phi_{i'}(\mathbf{x}^\mu)}{\partial u_j}, \tag{71}$$

and the second derivative is given by

$$\frac{\partial^2 \hat{y}_i^\mu}{\partial u_j \partial u_k} = \sum_{i'=1}^{n_{\mathcal{C}}} \left[\left((\delta_{i,i'} - \hat{y}_{i'}^\mu) \frac{\partial \hat{y}_i^\mu}{\partial u_k} - \hat{y}_i^\mu \frac{\partial \hat{y}_{i'}^\mu}{\partial u_k} \right) \frac{\partial \phi_{i'}(\mathbf{x}^\mu)}{\partial u_j} + \hat{y}_i^\mu (\delta_{i,i'} - \hat{y}_{i'}^\mu) \frac{\partial^2 \phi_{i'}(\mathbf{x}^\mu)}{\partial u_j \partial u_k} \right]. \tag{72}$$

Note, that we have expressed the derivatives as a function of the derivatives for a standard neural network with linear outputs: $\partial \phi_{i'}(\mathbf{x}^\mu)/\partial u_j$ and $\partial^2 \phi_{i'}(\mathbf{x}^\mu)/\partial u_j \partial u_k$.

It is often desirable for computational reasons to use the Gauss-Newton approximation of the Hessian instead,

$$\frac{\partial^2 E_{\mathcal{D}}(\mathbf{u})}{\partial u_j \partial u_k} \approx \frac{1}{q_{\mathcal{D}}} \sum_{\mu=1}^{q_{\mathcal{D}}} \sum_{i=1}^{n_{\mathcal{C}}} \frac{1}{\hat{y}_i^\mu} \frac{\partial \hat{y}_i^\mu}{\partial u_k} \frac{\partial \hat{y}_i^\mu}{\partial u_j}. \tag{73}$$

This is motivated by Fisher's property, $\left\langle \partial^2 e(\mathbf{x}, \mathbf{y}, \mathbf{u}) / \partial \mathbf{u} \partial \mathbf{u}^{\mathrm{T}} \right\rangle_{p(\mathcal{D})} = \left\langle \partial e(\mathbf{x}, \mathbf{y}, \mathbf{u}) / \partial \mathbf{u} \; \partial e(\mathbf{x}, \mathbf{y}, \mathbf{u}) / \partial \mathbf{u}^{\mathrm{T}} \right\rangle_{p(\mathcal{D})}$ [48], that is valid when using a log-likelihood cost function. An important property of this approximation is that the Hessian is guaranteed to be positive semi-definite, thus ensuring that a Newton step is a descent direction. The Newton algorithm will be described shortly.

The detailed derivations of equation (69)-(73) may be found in [49].

4.2.1 Gradient descent optimization

One of the simplest optimization algorithms is *gradient descent* also known as *steepest descent* derived from a first order Taylor approximation to the regularized cost function. It is based on iteratively updating the weight vector so that we move in the direction of the largest rate of decrease of the cost function, i.e., in the direction of the negative gradient of the cost function evaluated at timestep t. This may be written as

$$\Delta \mathbf{u}^{(t)} = -\eta \frac{\partial C(\mathbf{u}^{(t)})}{\partial \mathbf{u}} = -\eta \left(\frac{\partial E_{\mathcal{D}}(\mathbf{u}^{(t)})}{\partial \mathbf{u}} + \frac{\partial R(\mathbf{u}^{(t)})}{\partial \mathbf{u}} \right). \tag{74}$$

where η is called a *learning rate* that ensures that the cost error decreases for each iteration when η is sufficiently small.

It is clear that a too small learning rate will result in slow convergence while a too large learning rate will yield the first order approximation inadequate which may result in an error increase. A simple approach for iteratively adapting the learning rate is described in the following first order optimization scheme with fixed regularization parameters:

1. Initialize weights, e.g., uniformly over $[-0.5; 0.5]$.
2. Compute $C(\mathbf{u}^{(t)})$, initialize[14] the learning rate, η, and compute the weight parameter change, $\Delta \mathbf{u}^{(t)} = -\eta \partial C(\mathbf{u}^{(t)})/\partial \mathbf{u}^{(t)}$.
3. Update the weights, $\mathbf{u}^{(t+1)} = \mathbf{u}^{(t)} + \Delta \mathbf{u}^{(t)}$, and compute $C(\mathbf{u}^{(t+1)})$.
4. If $C(\mathbf{u}^{(t+1)}) > C(\mathbf{u}^{(t)})$, then set $\eta = \eta/2$ and goto step 3.
5. If the convergence criteria[15] is not met, then set $t = t + 1$ and goto step 2.

This simple gradient descent scheme is not very efficient but it may be employed when more sophisticated optimization schemes are not applicable. This could, e.g., be the case in the startup phase for optimization algorithms based on a second order Taylor expansion where the quadratic approximation initially may be poor. That is, the gradient descent algorithm may be applied as initialization for more advanced optimization schemes.

4.2.2 Newton optimization

There are several optimization algorithms based on a second order Taylor expansion of the cost function. One of these is the *Newton* optimization method [43].

Using a second order Taylor expansion of the regularized cost function, the weights are updated by,

$$\mathbf{u}^{(t+1)} = \mathbf{u}^{(t)} - \eta \left(\frac{\partial^2 E_{\mathcal{D}}(\mathbf{u}^{(t)})}{\partial \mathbf{u} \partial \mathbf{u}^{\mathrm{T}}} + \frac{\partial^2 R(\mathbf{u}^{(t)})}{\partial \mathbf{u} \partial \mathbf{u}^{\mathrm{T}}} \right)^{-1} \left(\frac{\partial E_{\mathcal{D}}(\mathbf{u}^{(t)})}{\partial \mathbf{u}} + \frac{\partial R(\mathbf{u}^{(t)})}{\partial \mathbf{u}} \right), \tag{75}$$

where η is a stepsize parameter that ensures a cost error decrease when the second order Taylor expansion is poor.

The Newton algorithm may be formulated as the following iterative scheme:

[14] Through initial experiments, a suitable value may be found.

[15] This could, e.g., be when the 2-norm of the gradient is below some small value.

1. Initialize weights, use, e.g., the gradient descent scheme in section 4.2.1.

2. Compute $C(\mathbf{u}^{(t)})$ and initialize the step size, η, to 1.

3. Update the weights according to eq. (75) and compute $C(\mathbf{u}^{(t+1)})$.

4. If $C(\mathbf{u}^{(t+1)}) > C(\mathbf{u}^{(t)})$, then set $\eta = \eta/2$ and goto step 3.

5. If the convergence criteria[16] is not met, then set $t = t + 1$ and goto step 2.

The Newton algorithm converges in very few iterations but may be computational expensive due to the need for computing and inverting the regularized Hessian.

4.3 Design algorithm overview

Based on the optimization algorithms described in this section and the probabilistic framework described in section 3, we will suggest a scheme for designing neural network classifiers based on adaptive estimation of the network architecture by using the optimal brain damage pruning technique described in section 3.4.2. The regularization parameters are fixed throughout the pruning scheme. The algebraic test error estimate described in section 3.3.2 is used for selection of the optimal network architecture.

In brief, the algorithm may be described as:

1. Determine the regularization parameters. These may be found, e.g., by sampling the algebraic test error estimate as a function of these parameters and choose those that minimize the algebraic test error estimate. An example of this is shown in [50].

2. Train/retrain the network using the Newton optimization algorithm. After pruning a small percentage of weights, only a few retraining iterations are usually required.

3. Compute the algebraic test error estimate.

4. Compute the OBD saliencies and remove a small percentage of the weights with the smallest saliencies. Goto 2, if any weights are left.

5. Select the network with the smallest algebraic test error estimate as the optimal network.

After designing a classifier using this algorithm, Bayes minimum-risk decision rule and rejection thresholds may be applied.

Examples of using the algorithm are shown in [51], [52].

[16] This could, e.g., be when the 2-norm of the gradient is below some small value.

5 EXPERIMENTS

We employ the design algorithm described in section 4.3 using fixed values of the regularization parameters combined with network pruning. Of particular interest is the pruning of dermatoscopic input features.

5.1 Experimental setup

We have a total of 58 dermatoscopic images distributed in 3 skin lesion categories as: *Benign nevi*: 25, *atypical nevi*: 11 and *malignant melanoma*: 22. For each image, 9 features have been extracted. In summary, these are: 2 asymmetry measures, 2 edge abruptness measures and 5 color measures (see section 2 for details).

One approach for attempting to overcome the limited data problem, would be to employ bootstrapping methods for increasing the training set size, see e.g. [53], [54], [55].

We will use the empirical leave-one-out test error estimator described in section 3.3.1 for evaluating the designed classifiers. This gives us 58 training sets each with 57 patterns and 58 test sets with 1 pattern. Thus, in order to design a complete classifier for solving the malignant melanoma problem, we need to design 58 classifiers for the 58 training sets.

The used network architecture consists of 9 inputs, 4 hidden units and 2 output units with 2 regularization parameters, α_{w^I} and α_{w^H}, for the weights/biases in the input layer and the weights/biases in the output layer, respectively.

The network weights are initialized uniformly over $[-0.5; 0.5]$ and the regularization parameters are set to $\alpha_{w^I} = 0.5$ and $\alpha_{w^H} = 0.9$. These are chosen in order to prevent significant overfitting of the training data. A more systematic approach for determining the regularization parameters without the use of a validation set is to sample the algebraic test error estimate as a function of the regularization parameters and use the regularization parameters that minimize the algebraic test error estimate. Examples of this are shown in [50]. 30 gradient descent iterations are performed prior to using the Newton algorithm[17] for locating a cost function minimum. Matrix inversion is done using the Moore-Penrose pseudo inverse (see e.g. [48]) ensuring that the eigenvalue spread[18] is less than 10^8. This is not a problem for this application due to the rather large regularization parameters.

Next, the network is pruned and the optimal pruned model is selected as the model with the lowest algebraic test error estimate. Recall, that this is an asymptotic estimate. Thus, its use may be questionable in an application with only 57 training patterns. During pruning, the *training patterns per weight* relationship improves, thus hopefully improving the validity of the estimator.

[17] Training is stopped when the 2-norm of the gradient of the training error w.r.t. the weights is below 10^{-5} or the maximum number of allowed iterations is reached.

[18] Eigenvalue spread should not be larger than the square root of the machine precision [56].

Table 2: Cross-entropy error for the malignant melanoma problem. The averages and standard deviations over 10 runs are reported. One run is a full leave-one-out scheme using 58 training sets.

Cross-entropy error	**Non-pruned neural classifier**	**Pruned neural classifier**
Training	0.689 ± 0.002	0.757 ± 0.003
Test	1.022 ± 0.016	1.007 ± 0.006

Table 3: Probability of misclassification for the malignant melanoma problem. The averages and standard deviations over 10 runs are reported.

Probability of misclassification	**Non-pruned neural classifier**	**Pruned neural classifier**
Training	0.273 ± 0.004	0.306 ± 0.001
Test	0.441 ± 0.023	0.400 ± 0.007

Since we employ the leave-one-out empirical test error estimator for model evaluation, the full classifier consists of 58 pruned networks.

5.2 Results

A total of 10 classifiers each consisting of 58 pruned networks as described in the previous section are designed. All results reported are the averages and standard deviations for the 10 classifiers.

5.2.1 Classifier results

Table 2 lists the cross-entropy error rates for the training and test set before and after pruning. As expected, the training error increases as a result of pruning due to the reduced network complexity while the test error decreases only slightly.

The corresponding classification[19] results are shown in table 3. Here we see a more noticeable decrease of the test error from 0.441 ± 0.023 to 0.400 ± 0.007 after pruning. Note, that there is still some discrepancy between the training error and test error suggesting that we are still overfitting the training set somewhat.

While the cross-entropy error and the classification error yield some insight into the performance of a classifier, it is of great interest to see how the classification errors are distributed in the 3 classes. This information is contained in *confusion matrices*.

[19] Following Bayes minimum-error decision rule as described in section 3.1, the network output with the highest probability determines the class. One could also adopt Bayes minimum-risk decision rule, see, e.g., [23].

Table 4: Confusion matrix for the test set using non-pruned networks. The averages and standard deviations over 10 runs are reported.

Confusion matrix	**Non-pruned neural classifier**		
for test set	**Benign nevi**	**Atypical nevi**	**Melanoma**
Benign nevi†	0.684 ± 0.058	0.709 ± 0.038	0.273 ± 0.000
Atypical nevi†	0.108 ± 0.033	0.018 ± 0.038	0.041 ± 0.014
Melanoma†	0.208 ± 0.041	0.273 ± 0.000	0.686 ± 0.014

† indicates the estimated output classes.

In table 4 and 5, the confusion matrices for the test set before and after pruning are shown. We see that the performance for the *atypical nevi* class is rather poor before pruning and even worse after pruning. The reason, that the *atypical nevi* class suffers, is the lower class prior[20] compared to the *benign nevi* and *melanoma* class. Thus, the error contribution from the *atypical nevi* class is relatively small making it fairly inexpensive to ignore this class during training. A method for minimizing the risk of completely ignoring a class is to weight each error contribution from a pattern in the cross-entropy error function with the inverse class prior. This corresponds to creating equal class priors. In order to take the real imbalanced priors into account, the network outputs should be reweighted with the real imbalanced class priors divided by the balanced class priors (see, e.g., [23]). This approach has not been employed in this work. It is interesting to note that the majority of the *atypical nevi* before and after pruning are assigned to the *benign nevi* class when recalling that the *atypical nevi* are in fact healthy. $72.7\% \pm 0.0\%$ are actually classified as benign for the pruned classifiers. This suggests that the information in the extracted dermatoscopic features is not adequate for distinguishing the *benign nevi* from the *atypical nevi* but is more appropriate for separating healthy lesions, i.e. *benign nevi* and *atypical nevi*, from cancerous lesions. Acknowledging this, we might be able to obtain a higher detection of the *melanoma* lesions by considering only these two categories of lesions when designing the classifiers. This has not been attempted, though. If we compare the test set results before and after pruning, we note that pruning has improved the detection of the *benign nevi* and the *melanoma* lesions significantly. In fact, a detection rate of $75.0\% \pm 2.4\%$ for the *melanoma* lesions are comparable with the detection rates of very experienced dermatologists [2].

In figure 12, the results of a typical run of the design algorithm is shown. For the non-pruned networks, the cross-entropy test error and classification test error exhibit only very little overfitting. Notice, how the Newton optimization sets in after 30 iterations. If smaller regularization parameters were used, the effects would have been a lot more dramatic. The pruning plots show that the decrease of the cross-

[20] Recall, only 11 of 58 lesions in the training set are atypical.

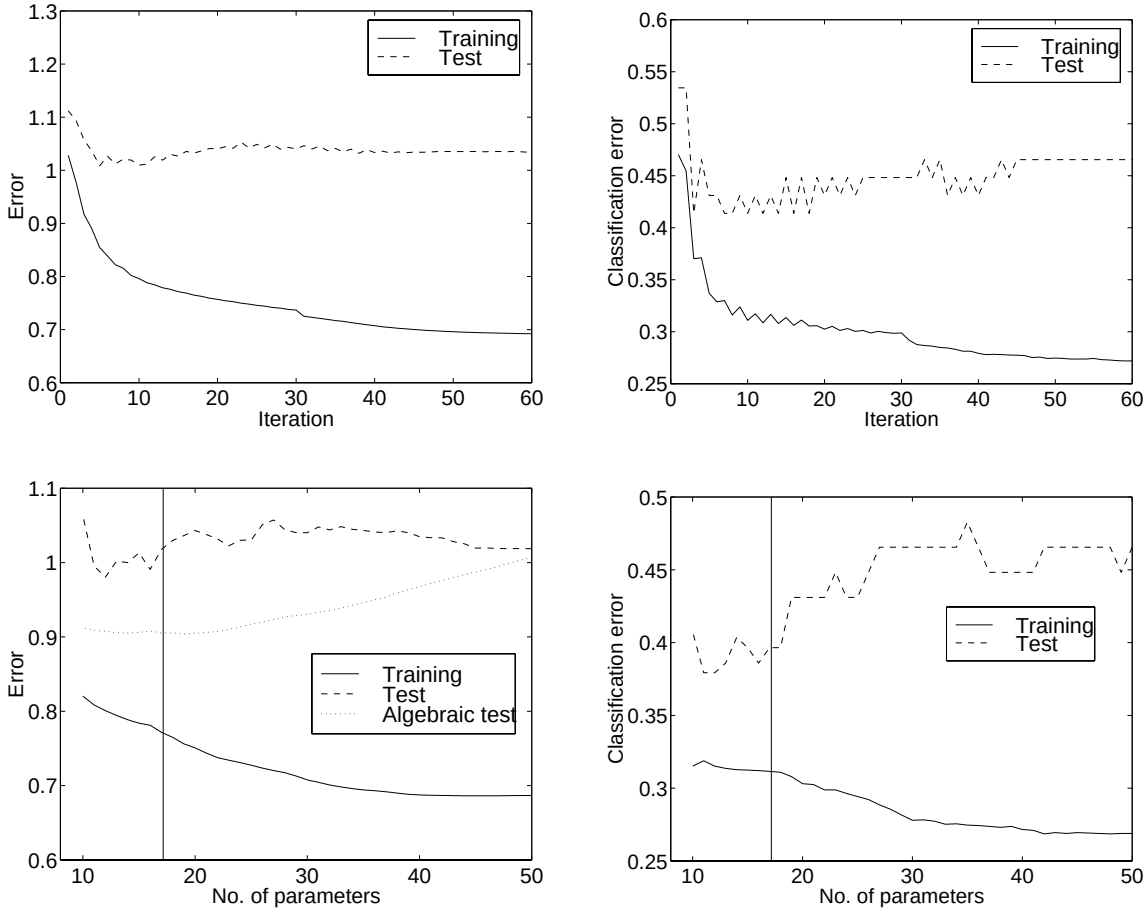

Figure 12: Results of a run of the design algorithm for the malignant melanoma problem. Each run consists of 58 networks. Upper left: The development of the cross-entropy error during training of the non-pruned networks. Gradient descent is used for the first 30 iterations, thereafter Newton optimization is used. Upper right: The development of the classification error during training of the non-pruned networks. Lower left: The development of the cross-entropy error during pruning. The vertical line indicates the mean location of the minimum of the estimated test error. Lower right: The development of the classification error during pruning.

Table 5: Confusion matrix for the test set using pruned networks. The averages and standard deviations over 10 runs are reported.

Confusion matrix	**Pruned neural classifier**		
for test set	**Benign nevi**	**Atypical nevi**	**Melanoma**
Benign nevi†	0.732 ± 0.019	0.727 ± 0.000	0.241 ± 0.037
Atypical nevi†	0.032 ± 0.017	0.000 ± 0.000	0.009 ± 0.019
Melanoma†	0.236 ± 0.013	0.273 ± 0.000	0.750 ± 0.024

† indicates the estimated output classes.

entropy test error and classification test error occurs at the end of the pruning session, i.e., when only 12 to 20 weights remain. Note, that the minimum of the algebraic test error estimate coincides fairly well with the region where the test error is lowest.

For comparison a standard *k-nearest-neighbor*[21] (k-NN) classification was performed. The training error may be computed from the training set by including each training pattern in the majority vote. The *leave-one-out* test error is computed by excluding each training pattern from the vote. Figure 13 shows the classification error on the training and test set as a function of k. We see that for a wide range of k-values, the k-NN classifier has similar classification error rates on the test set compared with the non-pruned and pruned neural classifiers suggesting that the k-NN classifier and the neural classifiers perform similarly. If we inspect the confusion matrix for the test set for a 15-NN classifier shown in table 6, we see that they classify quite differently despite having approximately the same overall classification error rate. The 15-NN classifier performs much better for the *benign nevi* class at the expense of the *melanoma* class. This is very unfortunate since the cancerous lesions are our major concern. From a medical point of view, it is significantly more expensive classifying a cancerous lesion as healthy as is the opposite case. Again, we note that a large majority of the *atypical nevi* are classified as *benign nevi* supporting our earlier statement concerning the discriminating power of the extracted dermatoscopic features.

5.2.2 Dermatoscopic feature importance

One of the most interesting effects of pruning is that it may provide information about the importance of the input variables. This is of particular interest for this application where the discriminating power of the dermatoscopic features is still rather unclear. Figure 14 shows an example of a pruned network selected by the minimum of the algebraic test error estimate. Two inputs have been completely removed by the

[21] Within a k-NN, a pattern is classified according to a majority vote among its k nearest neighbors using the Euclidean metric, see, e.g., [22].

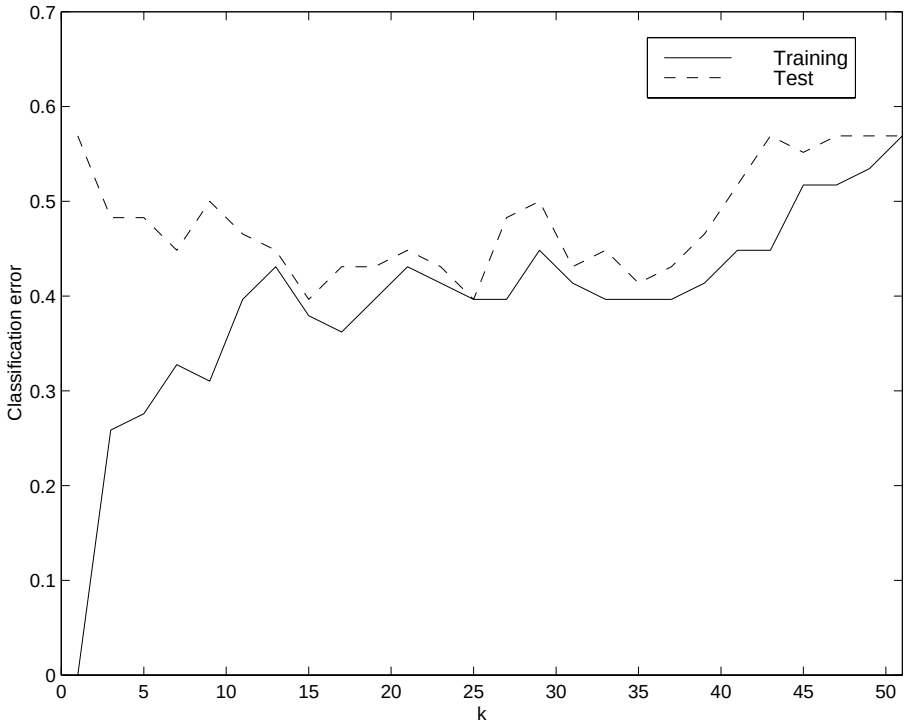

Figure 13: Classification results for a k-NN classifier as a function of k. Note, that for a wide range of k-values, the k-NN classifier performs similar to the non-pruned and pruned neural classifiers when comparing the classification rates.

Table 6: Confusion matrix for the test set using a 15-NN classifier. Note, that the classifier favors the *benign nevi* class, thus making costly errors in the *melanoma* class from a medical point of view.

Confusion matrix	**k-NN classifier ($k = 15$)**		
for test set	**Benign nevi**	**Atypical nevi**	**Melanoma**
Benign nevi†	0.920	0.818	0.455
Atypical nevi†	0.000	0.000	0.000
Melanoma†	0.080	0.182	0.545

† indicates the estimated output classes.

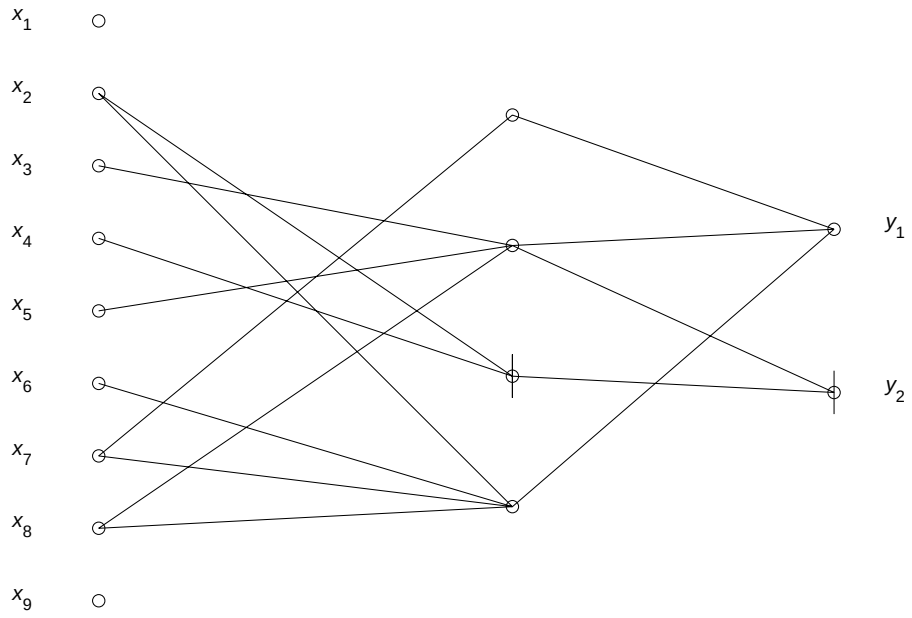

Figure 14: Example of a pruned malignant melanoma network with 17 weights. A vertical line through a node indicates a bias. The two pruned dermatoscopic input features are the *minor axis asymmetry* measure and the *dark-brown color* measure. These are the two most commonly pruned input features. Recall, that we only have two network outputs with weight connections due to the modified softmax normalization.

pruning process. For this particular network, it is the *minor axis asymmetry* measure and the *dark-brown color* measure. These are in fact the two most commonly removed dermatoscopic input features as can be seen in table 7. The table shows how often the individual dermatoscopic features have been completely removed during the runs of the design algorithm. Recall, that each run results in 58 pruned networks. Thus, for each run the number of times a feature has been removed is computed relative to the maximum number of times it could have been removed (58). This enables us to compute the mean and standard deviation over 10 runs and sort the features according to their importance[22]. Two features were never pruned: The *major axis asymmetry* measure and the *blue color* measure. We know that the presence of blue color in a lesion indicates *blue-white veil* and thus malignancy. So this is an expected result. We would also expect asymmetry to be important since this indicates different local growth rates in the lesion and thus malignancy. It is interesting to note that while the *major axis asymmetry* measure seems very important, the *minor axis asymmetry* measure is nearly always removed. The reason for this is probably that these two measures often are very similar which is also indicated by the skin lesion example in figure 6. That is, they both contain the same information, thus only one asymmetry measure is needed. The *dark-brown color* measure is the most often pruned feature. This is a bit surprising since the number of different colors present in a skin lesion normally is considered to correlate with the degree of malignancy. The removal of this feature could be due to the fact that the 5 color measures sum to 1 for a skin lesion. Thus, it is possible to infer a missing color measure from the remaining 4. We also note that the *white color* measure is often removed. This could invalidate the explanation of the inference of a missing color measure but the amount of white color, if present, is typically under 0.5%. That is, the *white color* measure could easily be ignored in the inference of the missing *dark-brown color* measure.

In summary, the 3 most important dermatoscopic features seem to be the *major axis asymmetry* measure and the *blue and black color* measures while the 3 least important are the *dark-brown and white color* measures and the *minor axis asymmetry* measure.

6 CONCLUSION

In this work, we have proposed a probabilistic framework for classification based on neural networks and we have applied the framework to the problem of classifying skin lesions.

This involved extracting relevant information from dermatoscopic images, defining a probabilistic framework, proposing methods for optimizing neural networks capable of estimating posterior class probabilities and applying the methods to the malignant melanoma classification problem.

[22] Assuming that the number of times a feature has been removed is inversely proportional to its importance.

Table 7: Table showing how often the individual dermatoscopic features have been completely pruned during the 10 runs. A zero pruning index for a feature indicates that it was never removed while a pruning index of 1 indicates that the feature was always removed. The averages and standard deviations over 10 runs are reported.

Feature importance	**Pruning index**	**Feature importance**	**Pruning index**	**Feature importance**	**Pruning index**
Asymmetry: Major axis	0.000 ± 0.000	**Edge abrupt.: Std. dev.**	0.053 ± 0.025	**Color: White**	0.272 ± 0.031
Color: Blue	0.000 ± 0.000	**Edge abrupt.: Mean**	0.083 ± 0.021	**Asymmetry: Minor axis**	0.772 ± 0.048
Color: Black	0.022 ± 0.008	**Color: Light-brown**	0.097 ± 0.023	**Color: Dark-brown**	0.783 ± 0.054

Dermatoscopic feature extraction

The extraction of dermatoscopic features involved measuring the skin lesion asymmetry, the transition of pigmentation from the skin lesion to the surrounding skin and the color distribution within the skin lesion. The latter involved determining color prototypes by inspecting 2-D color histograms and by using knowledge of dermatologists color perception. No reliable red prototype color could be identified, though, partially due to a strong reddish glow of the dark-brown color in skin lesions. It was seen that some of the extracted dermatoscopic features singlehandedly showed potential for separating in particular the malignant lesions from the healthy lesions.

Probabilistic framework for classification

The defined probabilistic framework for classification included optimal decision rules, derivation of error functions, model complexity control and assessment of generalization performance.

Neural classifier modeling

The proposed schemes for designing neural network classifiers involved defining a two-layer feed-forward network architecture and evoking methods for optimizing the network weights and the network architecture. Traditionally, a standard softmax output normalization scheme is employed in order to ensure that model outputs may be interpreted as posterior probabilities. This normalization scheme has an inherent redundancy due to the property that the posterior probability output estimates sum to one. This redundancy is generally ignored and results in weight dependencies in the output layer and, thus, a sin-

gular unregularized Hessian matrix. In order to overcome this, a modified softmax output normalization scheme removing the redundancy has been suggested.

The malignant melanoma classification problem

The neural classifier framework was applied to the malignant melanoma classification problem using the extracted dermatoscopic features and results from histological analyzes of skin tissue samples. The adaptive estimation of regularization parameters and outlier probability was not employed due to the very limited amount of data available. Instead, optimal brain damage pruning and model selection using an algebraic generalization error estimate was employed. In a leave-one-out test set, we were able to detect $73.2\% \pm 1.9\%$ of benign lesions and $75.0\% \pm 2.4\%$ of malignant lesions. None of the atypical lesions were classified correct. We argued that this probably is due to the fact that the atypical lesion class has a small prior and thus is ignored during model estimation. $72.7\% \pm 0.0\%$ of the atypical lesions were classified as benign lesions. Recalling, that atypical lesions are in fact healthy indicates that the extracted dermatoscopic features are effective only for separating healthy lesions from cancerous lesions, i.e., the features do not possess adequate information for discriminating between benign and atypical lesions. As a result of the pruning process, it was possible to rank the dermatoscopic features according to their importance. We found that the three most important features are shape asymmetry and the amount of blue and black color present within a skin lesion.

References

[1] B. Lindelöf and M.A. Hedblad. Accuracy in the Clinical Diagnosis and Pattern of Malignant Melanoma at a Dermatologic Clinic. *The Journal of Dermatology*, 21(7):461–464, 1994.

[2] H.K. Koh, R.A. Lew, and M.N. Prout. Screening for Melanoma/Skin Cancer. *Journal of American Academy of Dermatology*, 20(2):159–172, 1989.

[3] A. Østerlind. *Malignant Melanoma in Denmark*. PhD thesis, Danish Cancer Registry, Institute of Cancer Epidemiology, Denmark , 1990.

[4] G. Rassner. Früherkennung des malignen Melanoms der Haut. *Hausartz*, 39:396–401, 1988.

[5] Z.B. Argenyi. Dermoscopy (Epiluminescence Microscopy) of Pigmented Skin Lesions. *Dermatologic Clinics*, 15(1):79–95, January 1997.

[6] S. Fischer, P. Schmid, and J. Guillod. Analysis of Skin Lesions with Pigmented Networks. In *Proceedings of the International Conference on Image Processing*, volume 1, pages 323–326, 1996.

[7] P. Schmid and S. Fischer. Colour Segmentation for the Analysis of Pigmented Skin Lesions. In *Proceedings of the Sixth International Conference on Image Processing and its Applications*, volume 2, pages 688–692, 1997.

[8] H. Ganster, M. Gelautz, A. Pinz, M. Binder, P. Pehamberger, M. Bammer, and J. Krocza. Initial Results of Automated Melanoma Recognition. In G. Borgefors, editor, *Proceedings of The 9th Scandinavian Conference on Image Analysis*, pages 209–218, 1995.

[9] A. Steiner, M. Binder, M. Schemper, K. Wolff, and H. Pehamberger. Statistical Evaluation of Epiluminescence Microscopy Criteria for Melanocytic Pigmented Skin Lesions . *Journal of the American Academy of Dermatology* , 29(4):581–588, 1993.

[10] W. Stolz, O. Braun-Falco, P. Bilek, M. Landthaler, and A.B. Cognetta. *Color Atlas of Dermatoscopy.* Blackwell Science, Oxford, England, 1994.

[11] I. Stanganelli, M. Burroni, S. Rafanelli, and L. Bucchi. Intraobserver Agreement in Interpretation of Digital Epiluminescence Microscopy. *Journal of the American Academy of Dermatology*, 33(4):584–589, 1995.

[12] H. Karhunen. Über Lineare Methoden in der Wahrscheinlichkeitsrechnung. *American Academy of Science*, 37:3–17, 1947.

[13] M. Lòeve. Fonctions Aleatoires de Seconde Ordre. In P. Levy, editor, *Processus Stochastiques et Mouvement Brownien.* Hermann, 1948.

[14] K.V. Mardia, J.T. Kent, and J.M. Bibby. *Multivariate Analysis.* Academic Press, London, 1979.

[15] T.W. Ridler and S. Calvard. Picture Thresholding using an Iterative Selection Method. *IEEE Transactions on Systems, Man and Cybernetics*, 8(8):630–632, 1978.

[16] M. Sonka, V. Hlavac, and R. Boyle. *Image Processing, Analysis and Machine Vision.* Chapman & Hall, London, 1993.

[17] A.K. Jain. *Fundamentals of Digital Image Processing.* Prentice-Hall, New Jersey, 1989.

[18] B.D. Ripley. *Pattern Recognition and Neural Networks.* Cambridge University Press, Cambridge, 1996.

[19] A.J. Scott and M.J. Symons. Clustering Methods based on Likelihood Ratio Criteria. *Biometrics*, 27:387–397, 1971.

[20] W. Skarbek and A. Koschan. Colour Image Segmentation - A Survey. Technical Report 94-32, Institute for Technical Informatics, Technical University of Berlin, Germany, 1994.

[21] G. Wyszecki and W.S. Stiles. *Color Science.* Wiley, New York, 1982.

[22] R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis.* Wiley-Interscience, New York, 1973.

[23] C.M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, Oxford, 1995.

[24] M. Hintz-Madsen, L.K. Hansen, J. Larsen, E. Olesen, and K.T. Drzewiecki. Design and Evaluation of Neural Classifiers - Application to Skin Lesion Classification. In F. Girosi, J. Makhoul, E. Manolakos, and E. Wilson, editors, *Proceedings of the 1995 IEEE Workshop on Neural Networks for Signal Processing V*, pages 484–493, New York, New York, 1995.

[25] L.K. Hansen, C. Liisberg, and P. Salamon. The Error-reject Tradeoff. *Open Systems & Information Dynamics*, 4:159–184, 1997.

[26] D.J.C. MacKay. A Practical Bayesian Framework for Backpropagation Networks. *Neural Computation*, 4(3):448–472, 1992.

[27] D.J.C. MacKay. The Evidence Framework Applied to Classification Networks. *Neural Computation*, 4(5):720–736, 1992.

[28] H.H. Thodberg. Ace of Bayes: Application of Neural Networks with Pruning. Technical Report 1132E, The Danish Meat Research Institute, DK-4000, Denmark, 1993.

[29] S. Duane, A.D. Kennedy, B.J. Pendleton, and D. Roweth. Hybrid Monte Carlo. *Physics Letter B*, 2(195):216–222, 1987.

[30] R.M. Neal. *Bayesian Learning for Neural Networks.* PhD thesis, University of Toronto, Canada, 1994.

[31] J. Larsen and L.K. Hansen. Empirical Generalization Assessment of Neural Network Models. In F. Girosi, J. Makhoul, E. Manolakos, and E. Wilson, editors, *Proceedings of the 1995 IEEE Workshop on Neural Networks for Signal Processing V* , pages 30–39, New York, New York, 1995.

[32] M. Stone. Cross-validatory Choice and Assesment of Statistical Predictors. *Journal of the Royal Statistical Society*, 36(2):111–147, 1974.

[33] G.T. Toussaint. Bibliography on Estimation of Misclassification. *IEEE Transactions on Information Theory*, 20(4):472–479, 1974.

[34] L.K. Hansen and J. Larsen. Linear Unlearning for Cross-Validation. *Advances in Computational Mathematics*, 5:269–280, 1996.

[35] P.H. Sørensen, M. Nørgård, L.K. Hansen, and J. Larsen. Cross-Validation with LULOO. In S.I. Amari, L. Xu, I. King, and K.S. Leung, editors, *Proceedings of 1996 International Conference on Neural Information Processing*, volume 2, pages 1305–1310, Hong Kong, 1996.

[36] N. Murata, S. Yoshizawa, and S. Amari. A Criterion for Determining the Number of Parameters in an Artificial Neural Network Model. In *Artificial Neural Networks*, pages 9–14. Elsevier, Amsterdam, 1991.

[37] S. Amari and N. Murata. Statistical Theory of Learning Curves under Entropic Loss Criterion. *Neural Computation*, 5:140–153, 1993.

[38] N. Murata, S. Yoshizawa, and S. Amari. Network Information Criterion - Determining the Number of Hidden Units for an Artificial Neural Network Model. *IEEE Transactions on Neural Networks*, 5:865–872, 1994.

[39] H. Akaike. A New Look at the Statistical Model Identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974.

[40] L. Ljung. *System Identification: Theory for the User*. Prentice-Hall, New Jersey, 1987.

[41] J.E. Moody. The Effective Numbers of Parameters: An Analysis of Generalization and Regularization in Nonlinear Models. In J.E. Moody, S.J. Hanson, and R. Lippmann, editors, *Advances in Neural Information Processing Systems*, volume 4, pages 847–854, San Mateo, California, 1992.

[42] J. Larsen. *Design of Neural Network Filters*. PhD thesis, Electronics Institute, Technical University of Denmark, 1993.

[43] J. Hertz, A. Krogh, and R.G. Palmer. *Introduction to the Theory of Neural Computation*. Addison-Wesley, Reading, Massachusetts, 1991.

[44] A.E. Hoerl and R.W. Kennard. Ridge Regression. *Technometrics*, 12:55–82, 1970.

[45] Y. Le Cun, J. Denker, and S. Solla. Optimal Brain Damage. *Advances in Neural Information Processing Systems*, 2:598–605, 1990.

[46] J. Park and I.W. Sandberg. Universal Approximation using Radial-basis-function Networks. *Neural Computation*, 3:246–257, 1991.

[47] J.S. Bridle. Probabilistic Interpretation of Feedforward Classification Network Outputs with Relationships to Statistical Pattern Recognition. In F. Fougelman-Soulie and J. Herault, editors, *Neurocomputing - Algorithms, Architectures and Applications*, volume 6, pages 227–236. Springer-Verlag, Berlin, 1990.

[48] G.A.F. Seber and C.J. Wild. *Nonlinear Regression.* John Wiley & Sons, New York, New York, 1995.

[49] M. Hintz-Madsen. *A Probabilistic Framework for Classification of Dermatoscopic Images.* PhD thesis, Department of Mathematical Modelling, Technical University of Denmark, DK-2800 Lyngby, Denmark, 1998.

[50] M. Hintz-Madsen, L.K. Hansen, J. Larsen, M.W. Pedersen, and M. Larsen. Neural Classifier Construction using Regularization, Pruning and Test Error Estimation. *Neural Networks*, In press, 1998.

[51] M. Hintz-Madsen, L.K. Hansen, J. Larsen, E. Olesen, and K.T. Drzewiecki. Detection of Malignant Melanoma using Neural Classifiers. In A.B. Bulsari, S. Kallio, and D. Tsaptsinos, editors, *Solving Engineering Problems with Neural Networks - Proceedings of the International Conference on Engineering Applications of Neural Networks (EANN'96)*, pages 395–398, Turku, Finland, 1996.

[52] M. Hintz-Madsen, M.W. Pedersen, L.K. Hansen, and J. Larsen. Design and Evaluation of Neural Classifiers. In S. Usui, Y. Tohkura, S. Katagiri, and E. Wilson, editors, *Proceedings of the 1996 IEEE Workshop on Neural Networks for Signal Processing VI*, pages 223–232, New York, New York, 1996.

[53] G.A. Young. Bootstrap: More than a Stab in the Dark? *Statistical Science*, 9(3):382–415, 1994.

[54] B. Efron and R. Tibshirani. Bootstrap Methods for Standard Errors, Confidence Intervals, and Other Measures of Statistical Accuracy. *Statistical Science*, 1(1):54–77, 1986.

[55] B. Efron and R.J. Tibshirani. *An Introduction to the Bootstrap.* Monographs on Statistics and Applied Probability. Chapman & Hall, 1993.

[56] J.E. Dennis and R.B. Schnabel. *Numerical Methods for Unconstrained Optimization and Non-linear Equations.* Prentice-Hall, Englewood Cliffs, New Jersey, 1983.