In: Konstantin G. Korotkov (Ed.). Measuring Energy Fields: Current Research. – Backbone Publishing Co. Fair Lawn, USA, 2004.pp. 193-208.

CLASSIFICATION OF DIFFERENT TYPES OF CORONAS USING PARAMETRIZATION OF IMAGES AND MACHINE LEARNING

Igor Kononenko, Matjaz Bevk, Sasa Sadikov, Luka Sajn

University of Ljubljana, Faculty of Computer and Information Science,

Ljubljana, Slovenia, e-mail: igor.kononenko@fri.uni-lj.si

Abstract

We describe the development of computer classifiers for various types of coronas. In particular, we were interested to develop an algorithm for detection of coronas of people in altered states of consciousness (two-classes problem). Such coronas are known to have rings (double coronas), special branch-like structure of streamers and/or curious spots. Besides detecting altered states of consciousness we were interested also to classify various types of coronas (six-classes problem).

We used several approaches to parametrization of images: statistical approach, principal component analysis, association rules and GDV software approach extended with several additional parameters. For the development of the classifiers we used various machine learning algorithms: learning of decision trees, naïve Bayesian classifier, K-nearest neighbors classifier, Support vector machine, neural networks, and Kernel Density classifier. We compared results of computer algorithms with the human expert's accuracy (about 77% for the two-classes problem and about 60% for the six-classes problem). Results show that computer algorithms can achieve the same or even better accuracy than that of human experts (best results were up to 85% for the two-classes problem and up to 65% for the six-classes problem).

1. Introduction

Recently developed technology by dr. Korotkov (1998) from Technical University in St.Petersburg, based on the Kirlian effect, for recording the human bio-electromagnetic field (aura) using the Gas Discharge Visualization (GDV) technique provides potentially useful information about the biophysical and/or psychical state of the recorded person. In order to make the unbiased decisions about the state of the person we want to be able to develop the computer algorithm for extracting information/describing/classifying/making decisions about the state of the person from the recorded coronas of fingertips.

The aim of our study is to differentiate 6 types of coronas, 3 types in normal state of consciousness: **Ia, Ib, Ic** (pictures were recorded with single GDV camera in Ljubljana, all with the same settings of parameters, classification into 3 types was done manually):

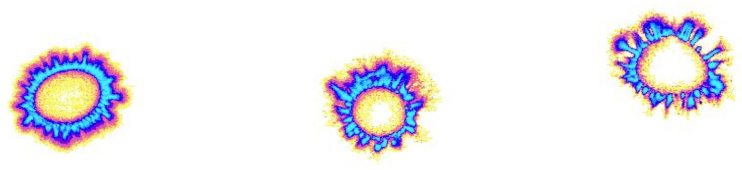
- **Ia** – harmonious energy state (120 coronas)
- **Ib** – non-homogenous but still energetically full (93 coronas)
- **Ic** – energetically poor (76 coronas)

and 3 types in altered states of consciousness (pictures obtained from dr. Korotkov, recorded by different GDV cameras with different settings of parameters and pictures were not normalized – they were of variable size):

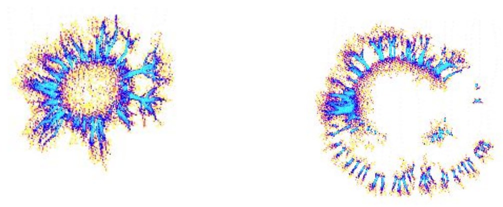
- **Rings** – double coronas (we added 7 pictures of double coronas recorded in Ljubljana) (90 coronas)
- **Branches** – long streamers branching in various directions (74 coronas)

- ❖ **Spots** – unusual spots (51 coronas)

Our aim is to differentiate normal from altered state of consciousness (2 classes) and to differentiate among all 6 types of coronas (6 classes). Figure 1 provides example coronas for each type.



Types Ia, Ib and Ic– normal state of consciousness



Types Branches, Rings and Spots– altered states of consciousness

Figure 1: Example coronas for each type.

2. The methodology

We first had to preprocess all the pictures so that all were of equal size (320 x 240). We then described the pictures with various sets of numerical parameters (attributes) with five different parametrization algorithms (described in more detail in the next section):

a) IP (Image Processor – 22 attributes),
b) PCA (Principal Component Analysis),
c) Association Rules,
d) GDV Assistant with some basic GDV parameters,
e)GDV Assistant with additional parameters.

Therefore we had available 5 different learning sets for two-classes problem: altered (one of Rings, Spots, and Branches) versus non- altered (one of Ia, Ib, Ic) state of consciousness. Some of the sets were used also as six-classes problems (differentiating among all six different types of coronas).

We tried to solve some of the above classification tasks by using various machine learning algorithms as implemented in Weka system (Witten and Frank, 2000):

- Quinlan's (1993) C4.5 algorithm for generating decision trees;
- K-nearest neighbor classifier by Aha, D., and D. Kibler (1991);
- Simple Kernel Density classifier;
- Naïve Bayesian classifier using estimator classes: Numeric estimator precision values are chosen based on analysis of the training data. For this reason, the classifier is not an Updateable Classifier (which in typical usage are initialized with zero training instances, see (John and Langley, 1995));
- SMO implements John C. Platt's sequential minimal optimization algorithm for training a support vector classifier using polynomial kernels. It transforms the output of SVM

into probabilities by applying a standard sigmoid function that is not fitted to the data. This implementation globally replaces all missing values and transforms nominal attributes into binary ones (see Platt, 1998; Keerthi et al., 2001);
- Neural networks: standard multilayared feedforward neural network with backpropagation of errors learning mechanism (Rumelhart et al., 1986).

SMO algorithm can be used only for two-classes problems, while the other algorithms can be used on two-classes and on six-classes problems.

3. Description of different parametrization procedures

3.1 Image Processor: First and second-order statistics on textures

IP (Image Processor) is a program, developed by Bevk (Bevk and Kononenko, 2002) for description of textures that implements a set of parameters as described by Julesz et al. (1973). Texture is a very commonly used term in computer vision. We all recognize texture when we see it, but it is very difficult to define it precisely. The main question is: "When is a texture pair distinguishable (by human), given that they have the same brightness, contrast and color?" Julesz et al. (1973) have studied human texture perception in the context of its discrimination. Their work concentrates on the spatial statistics of texture gray levels. To summarize their work, we need to define the first- and second-order spatial statistics:
- *First-order statistics* are calculated from the probability of observing a particular pixel value at a randomly chosen location in the image. They depend only on individual pixel values and not on interaction of neighboring pixel values. The mean of image grey intensity is an example of the first-order statistic.
- *Second-order statistics* are calculated from the probability of observing a pair of pixel values in the image that are some vector $\vec{d}$ apart. Note that second-order statistics become first-order if $\vec{d} = (0,0)$.

3.1.1 First-order statistics

First-order statistics are quite straightforward. They are computed from a function that measures the probability of a certain pixel occurring in an image. This function is also known as histogram. Histogram on grey scale images is defined as follows:

$$H(g) = \frac{n_g}{N}; g = 0,1,...,G-1$$

Where N is the number of all pixels in an image, G is the number of grey levels and n_g is the number of pixels of value g in an image. As we can see H is a probability function of pixel values, therefore we can characterize its properties with a set of statistical parameters (also called first-order statistics). Below is a list of such parameters, where represents n-th parameter of H.
- Mean (average brightness)
- Standard deviation
- Skew

$$P_3 = \frac{\sum_{g=0}^{G-1} (g-P_1)^3 H(g)}{P_2^3}$$

- Energy

$$P_4 = \sum_{g=0}^{G-1} H^2(g)$$

- Entropy

$$P_5 = -\sum_{g=0}^{G-1} H(g) \log_2 (H(g))$$

- N-th moment

$$P_{5+n} = \sum_{g=0}^{G-1} (g-P_6)^n H(g), n = 1,2,3,4$$

3.1.2 Second-order statistics

Second-order statistics operate on probability function that measures the probability of a pair of pixel values occurring some vector $\vec{d}$ apart in the image. This probability function is also called *co-occurrence matrix (CM)*, since it measures the probability of co-occurrence of two pixel values. Let us now define CM on the grey scale image. Suppose we have an $M \times N$ image with color space of G grey levels. Given a certain displacement vector $+\vec{d} = (dx, dy)$ the CM looks like this

$$C_{+\vec{d}}(i,j) = \frac{\left| \left\{ ((r,s),(t,v)); \begin{array}{r} (r,s)=i,(t,v)=j, \\ (r,s),(t,v) \in M \times N, \\ (t,v)=(r+dx,s+dy) \end{array} \right\} \right|}{(M-dx)(N-dy)}$$

Note that so defined CM is not symmetrical in terms of

$$C_{+\vec{d}}(i,j) \neq C_{+\vec{d}}(j,i)$$

A symmetrical CM can be computed using the formula

$$C_{\vec{d}} = \frac{1}{2}\left(C_{+\vec{d}} + C_{-\vec{d}}\right)$$

Suppose we have a set of displacement vectors $\Delta = \{\vec{d}_1, \vec{d}_2, ..., \vec{d}_n\}$. The final CM C used in second-order statistics is then the average at certain grey level over all CMs based on displacement vectors from .

Mostly used set of distance vectors is . This actually means that we are considering four neighboring pixels. It is empirically proven that this is a rather successful set, but there is no proof that this is a universally good set. This set holds pretty good information on local (short distance) pixel dependencies, but lacks of global (long distance) pixel dependencies. The more we extend with vectors that represent larger distances, the more local information we

loose. This loss of information happens because of the averaging in the last step of calculation of CM. There is obviously a trade off between local and global information that CM could hold.
In order to describe CM C of grey scale image Haralick et al. (1973) proposed a set of second-order statistics. These statistics are listed below. In the list F_n represents n-th statistics of C.

- Energy

$$F_1 = \sum_{i=1}^{G-1}\sum_{j=1}^{G-1} C^2(i,j)$$

- Entropy

$$F_2 = -\sum_{i=0}^{G-1}\sum_{j=0}^{G-1} C(i,j)\log_2\left(C(i,j)\right)$$

- Contrast

$$F_3 = \sum_{i=0}^{G-1}\sum_{j=0}^{G-1} (i-j)^2\, C(i,j)$$

- Homogeneity

$$F_4 = \sum_{i=0}^{G-1}\sum_{j=0}^{G-1} \frac{C(i,j)}{1+(i-j)^2}$$

- Correlation
  Let us first define marginal probabilities of C:

$$C'(i) = \sum_{j=0}^{G-1} C(i,j) \quad C''(j) = \sum_{i=0}^{G-1} C(i,j)$$

  $C' = C''$, since C is symmetrical. Notice what is happening here:
  A probability of a certain $C'(i)$ is calculated so that probabilities over all possible pairs of i are summed. What we get is a probability of pixel value *i* itself. That is actually a histogram of an image. The same holds for C'' as well.
  Let $\mu_{C'}$ and $\mu_{C''}$ be the means of marginal probabilities

$$\mu_{C'} = \sum_{i=0}^{G-1} iC'(i) \quad \mu_{C''} = \sum_{j=0}^{G-1} jC''(j)$$

  Note that $\mu_{C'} = \mu_{C''}$. If C' and C'' are histograms of an image and $\mu_{C'}$ and $\mu_{C''}$ are their means, we can easily see what $\mu_{C'}$ and $\mu_{C''}$ represent. They simply represent the average pixel value of an image.
  Further let and be the standard deviations of C' and C'':

  Obviously . Now we are ready to define a measure of correlation for C, that is:

- Variance

$$F_6 = \sum_{i=0}^{G-1}\sum_{j=0}^{G-1}\left(i-\mu_{C'}\right)^2 C(i,j)$$

- Mean of pair-sums
  First define a new probability function, a function that represents a probability of a particular pair-sum of pixel values. Let's denote it with S:
  $$S(x) = \sum_{i+j=x} C(i,j)$$
  Now we can calculate their means
  $$F_7 = \sum_{x=0}^{2(G-1)} xS(x)$$
- Variance of pair-sums
  $$F_8 = \sum_{x=0}^{2(G-1)} \left(x-F_7\right)^2 S(x)$$
- Entropy of pair-sums
  $$F_9 = -\sum_{x=0}^{2(G-1)} S(x)\log_2\left(S(x)\right)$$
- Variance of pair-differences
  Like we defined a probability function for pair-sums we can also define a probability function for all possible pair-differences. We will denote it with D:
  $$D(x) = \sum_{|i-j|=x} C(i,j)$$
  Now it is possible to define a mean of pair-differences
  $$\mu_D = \sum_{x=0}^{G-1} xD(x)$$
  And finally we are able to calculate variances
  $$F_{10} = \sum_{x=0}^{G-1}\left(x-\mu_D\right)^2 D(x)$$
- Entropy of pair-differences
  $$F_{11} = -\sum_{x=0}^{G-1} D(x)\log_2\left(D(x)\right)$$
- Information measure of correlation 1
  $$F_{12} = \frac{F_2 + \sum_{i=0}^{G-1}\sum_{j=0}^{G-1} C(i,j)\log_2\left(C'(i)C''(j)\right)}{-\sum_{i=0}^{G-1} C'(i)\log_2\left(C'(i)\right)}$$
- Information measure of correlation 2

3.2 Association rules for texture description

We used the algorithm developed by Bevk (2003) and based on fast Apriori algorithm (Agrawal and Srikant, 1994) for generation of association rules in large databases. Association rules were developed by Agrawal et al. (1993). The following is a formal statement of the problem: Let $I = \{i_1, i_2, \ldots, i_m\}$ be a set of literals, called items. Let D be a set of transactions, where each transaction T is a set of items such that $T \subseteq I$. We say that a transaction T contains X, if $X \subseteq T$. An association rule is an implication of the form $X \Rightarrow Y$, where $X \subseteq I, Y \subseteq I, X \cap Y = 0$. The rule $X \Rightarrow Y$ holds in the transaction set D with *confidence* c if c% of transactions in D that contain X also contain Y. The rule $X \Rightarrow Y$ has *support* s in the transaction set D if s% of transactions in D contain $X \cup Y$.
The problem of discovering association rules says: Find all association rules in transaction set D with confidence of at least *minconf* and support of at least *minsup*, where minconf and minsup represent the lower boundary for confidence and support of association rules.

3.2.1 Texture representation in the context of association rules

The use of association rules for texture description was first introduced by Rushing et al. (2001). Here we present a slightly different approach, which uses different texture representation and different algorithm for association rules induction.
In order to use association rules algorithms on textures, we have to define transactions, transaction items, association rules and the transaction set in the context of textures.

A pixel $\vec{A}$ *of a texture P* is a vector $\vec{A} = (X, Y, I) \in P$, where X and Y represent absolute coordinates, and I represents the intensity of texture pixel.

A root pixel can be any chosen pixel a texture $\vec{K} = (X_K, Y_K, I_K)$.

R neighborhood $N_{R,\vec{K}}$ is a set of texture pixels, which are located in the circular area of radius R with root pixel $\vec{K}$ at the center. Root pixel $\vec{K}$ itself is not a member of its neighborhood $N_{R,\vec{K}}$.

$$N_{R,\vec{K}} = \left\{ (X, Y, I) \middle| \left\lfloor \sqrt{(X_K - X)^2 + (Y_K - Y)^2} + 0.5 \right\rfloor \leq R \right\} \setminus \vec{K}$$

Transaction $T_{R,\vec{K}}$ is a set of elements based on its corresponding neighborhood $N_{R,\vec{K}}$. The elements of transaction are represented with Euclidean distance and intensity difference from root pixel.

Transaction element is two-dimensional vector , where the first component represents Euclidean distance from root pixel and the second component represents intensity difference from root pixel.

Association rule is composed of transaction elements; therefore its form looks like this

Transaction set $D_{P,R}$ is composed of transactions, which are derived from all possible root pixels of a texture P at certain neighborhood size R:

$$D_{P,R} = \left\{ T_{R,\vec{K}} \middle| \forall \vec{K} : \vec{K} \in P \right\}$$

This representation of textures allows us to use general algorithms for association rules on textures. Also note that this representation is rotation invariant.

3.2.2 From association rules to feature description

Using association rules on textures, will allow to extract a set of features (attributes) for a particular domain of textures. Here is the general algorithm for that purpose:

- *Select a (small) subset of images F for feature extraction.*
  The subset F can be considerably small. Use at least one typical example of images in the domain. That is at least one sample per class or more, if the class consists of subclasses.
- *Pre-process the images in F.*
  Pre-processing involves the transformation of images to grey scale if necessary, the quantization of grey levels and the selection of proper neighborhood size R.
  The initial number of grey levels per pixel is usually 256. The quantization process downscales it to say 16 levels per pixel. Typical neighborhood sizes are 3, 4, 5.
- *Generate association rules form images in F.*
  Because of the representation of texture, it is possible to use any algorithm for association rules extraction. We use Apriori and GenRules as described in (Agrawal et al., 1993).
- *Use generated association rules to extract a set of features.*
  There are two attributes associated with each association rule: support and confidence. Use these two attributes of all association rules to construct a feature set. The number of extracted features is twice the number of association rules, which could be quite a lot. We therefore recommend using some feature selection algorithm, which can be of general purpose, namely the features are continuous values between 0 and 1.

3.2.3 Use the extracted feature set for classification

To use the extracted feature set for classification, calculate the features on the images excluded from feature extraction. This means calculating support and confidence of extracted association rules. The result is a data set with continuous features. The representation of images with a set of continuous features (attributes) allows us to choose from a large variety of machine learning algorithms for classification.

3.3 Principal component analysis (PCA)

Principal component analysis (PCA) involves a mathematical procedure that transforms a number of (possibly) correlated variables into a (smaller) number of uncorrelated variables called *principal components*. The first principal component accounts for as much of the variability in the data as possible, and each succeeding component accounts for as much of the remaining variability as possible.
The idea of using PCA for image processing was first used by Sirovich and Kirby (1987). Their work uses PCA for the image database compression. Also, a very well known work was done by Turk and Pentland (1991), who use PCA for human face recognition.
Let us summarize the face recognition algorithm. This method tries to find the principal components in the distribution of features across images, also called the eigenvectors. These

eigenvectors can be considered as a set of features, which together characterize the variation between images. Each of the images in the training set can be represented exactly as linear combination of the eigenvectors. If we use only some eigenvectors that have the largest eigenvalues, we can approximate and represent the most significant variation within the image set. Therefore the best M eigenvectors span a M-dimensional subspace of all possible images. Eigenvectors, also called characteristic vectors, are computed from the covariance matrix, which is composed of the differences between each image and the average image.

Here is more formal procedure of this algorithm. Every image $I_j; j \in [1..M]$, which is a $X \times Y$ matrix, is here represented as a $XY = N$ dimensional vector $\vec{i}_j$. Because we are interested in the variation between pictures, we calculate the average image:

$$\vec{c} = \frac{1}{M}\sum_{j=1}^{M}\vec{i}_j$$

Then we calculate for each image $\vec{i}_j$ its difference from average image $\vec{c}$:

$$\vec{d}_j = \vec{i}_j - \vec{c}, j \in [1..M]$$

From here on, each image $\vec{i}_j$ is represented with its difference $\vec{d}_j$ from average image. Now, let's compose a matrix of differences:

$$P = \left[\vec{d}_1, \vec{d}_2, .., \vec{d}_M\right]$$

The covariance matrix Q of the matrix P would be calculated as $Q = PP^T$ .Q is $N \times N$ matrix. Since N is usually a very large number, it is computationally expensive to compute eigenvectors from it. Luckily there is a way around that problem. First compute a (usually) smaller matrix:

$$\tilde{Q} = P^T P$$

Note that $\tilde{Q}$ is $M \times M$ matrix, which is usually smaller than $N \times N$, since the number of images is smaller than the number of pixels in each image. It can be shown that the M largest eigenvalues and the corresponding eigenvectors of Q can be determined from the M eigenvalues and eigenvectors of $\tilde{Q}$ (Turk and Pentland, 1991).

Having eigenvectors $\vec{e}\,'_j$ and eigenvalues λ'_j of $\tilde{Q}$, eigenvectors $\vec{e}_j$ and eigenvalues λ_j of Q can be calculated using these simple formulas:

$$\lambda_j = \lambda'_j$$

$$\vec{e}_j = \frac{1}{\sqrt{\lambda'_j}} P\vec{e}\,'_j$$

Next we project each difference vector onto subspace determined by largest M eigenvectors:

Each projection is called feature vector. In other words, holds a set of features describing the image .

The features of some new image are calculated so, that we project its difference vector onto subspace determined by largest eigenvectors. The projection represents a set of features describing this new image .

Let us go back to the problem of finding eigenvectors and eigenvalues of some matrix A. A matrix A is said to have an eigenvector x and corresponding eigenvalue λ if

$$Ax = \lambda x$$

The upper equation can also be formulated as

$$\left| A - \lambda I \right| = 0$$

which, if expanded out, is an Nth degree polynomial in lambda whose roots are eigenvalues. Obviously there are always N (not necessarily distinct) eigenvalues. Root searching in the upper equation is a very poor computational method.

In our case, we want to find eigenvectors of a covariance matrix $\tilde{Q}$, which is always symmetric. The procedure is as follows:

First we transform symmetric matrix $\tilde{Q}$ to tridiagonal form using *Householder reductions* (Press et al., 1992). Secondly we calculate eigenvalues and eigenvectors from tridiagonal matrix using *QL Algorithm* (Press et al., 1992).

3.4 GDV Assistant

GDV Assistant is the system developed by Sadikov (2002) in order to have a complete control of various numerical parameters, developed by Korotkov (1998) and in order to experiment with various additional parameters. Here we provide only a list of used numerical parameters. A more detailed description can be found in this volume (Sadikov and Kononenko, 2003).

3.4.1 Basic parameters of GDV Assistant

Sadikov (2002) reimplemented the calculation of ten numerical parameters, as implemented by software package GDV Analysis (Korotkov, 1998):
A1. Area of GDV-gram,
A2. Noise, deleted from the picture,
A3. Form coefficient I,
A4. Fractal dimension,
A5. Brightness coefficient,
A6. Brightness deviation,
A7. Number of separated fragments in the image,
A8. Average area per fragment,
A9. Deviation of fragments' areas,
A10. Relative are of GDV-gram.

Besides, we implemented also two parameters, defined by Korotkov and Korotkin (2001): *average streamer width* and *entropy of corona,* and also five additional parameters (Sadikov, 2002): *corona width, form deviation, normalized skewness of brightness*, *normalized stability of brightness*, and *entropy of brightness.* All parameters, with the exception of corona width, are described in (Sadikov et al., 2003, this volume). Corona width is defined as follows:
where *Area* is absolute area (A1) of corona and $S_{ellipse}$ is the area of inner oval (ellipse) of corona.

3.4.2 GDV Assistant with additional parameters

In order to get more information about fragments of corona and their spatial relations we developed ten additional parameters:
- average distance of fragment from corona center,
- st. dev. of distance of fragment from corona center,
- average distance of fragment from nearest neighbor,
- st. dev. of distance of fragment from nearest neighbor,

- average distance of fragment from corona ellipse,
- st. dev. of distance of fragment from corona ellipse,
- average number of edges per fragment,
- average length of fragments edge,
- st. dev. of edge points from fragment center,
- st. dev. of edge points from fragment center relatively to the fragment.

4. Results

4.1 Results for C4.5

In the first experiment we tried to learn decision trees from five different descriptions of data, as returned by different parametrization algorithms, for two-classes problem. We used Quinlan's C4.5 algorithm. For testing we used the standard 10-fold cross validation:
1. the whole training set is randomly divided into ten subsets of equal size,
2. in each run, one subset is kept for testing and the union of the remaining nine subsets is used for training the classifier,
3. the result (classification error) is averaged over ten runs and the standard error is calculated.

Results are given in Table 1. We provide
- the number of numerical parameters (attributes) used to describe the training instances,
- the achieved classification error,
- standard error which is equal to standard deviation divided with the square root of the number of testing instances, and
- default error which is equal to 100 minus the percentage of instances that belong to majority (most probable) class (for our data set 289 out of 504 = 57% of instances belong to class *normal state of consciousness*); default error shows how difficult is the classification problem, as default error can be achieved by very simple classifier which classifies every testing instance into the majority class.

Results show that GDV Assistant achieves best results, which was also expected, although we expected larger advantage over other parametrization algorithms. Additional attributes for description of different statistics of fragments did not provide any improvement, which is somehow disappointing.

	Number of attributes	Classification error	Standard error	Default error
IP	22	19.8 %	0.9 %	43,0 %
PCA	15	29.2 %	1.8 %	43,0 %
Assoc.rules	44	20.0 %	1.6 %	43,0 %
GDV Assist.	17	18.6 %	1.5 %	43,0 %
GDV Assist with add. atts	27	18.3 %	1.5 %	43,0 %

Table 1: Classification error of C4.5 on five different descriptions of coronas for two-class problem

4.2 Results of a human expert

In order to get a better feeling about how good are the above results in comparison to humans, we tested a human expert on one fold (51 testing instances) and compared his result with the accuracy of C4.5 (using the parametrization with Associative rules) on the same testing set. Results are as follows.

On the two-classes problem the human expert and C4.5 achieved the same classification error of 23.5%. The misclassification matrices are provided in Table 2. It seems that C4.5 is biased towards classifying more coronas as normal and therefore coronas for the altered state of consciousness are poorly classified. On the other hand the human expert does not have such bias and the misclassifications are more evenly distributed between two classes.

	human expert			C4.5	
	classified as (a)	classified as (b)		classified as (a)	classified as (b)
Class normal state (a)	22	4		28	1
Class altered state (b)	8	17		11	11

Table 2: Misclassification matrices for the two-classes problem of a human expert and C4.5

On the six-classes problem the human expert had classification error of 39,2%. C4.5 was significantly worse with 54.9%. The misclassification matrices are provided in Tables 3.1 and 3.2. Again it seems that C4.5 is biased towards classifying more coronas as normal and therefore coronas for the altered state of consciousness are poorly classified. On the other hand the human expert does not have such bias and the misclassifications are more evenly distributed among normal an altered states of consciousness.

The easiest types of coronas for classification are Ia (normal) and Rings (altered). The most difficult type of coronas seems to be Branches (altered), which is by human expert most often confused with Ib (normal) and Spots (altered), and by C4.5 with Ic (altered).

	classified as (a)	classified as (b)	classified as (c)	classified as (d)	classified as (e)	classified as (f)
Class normal Ia (a)	9	2	1			
Class normal Ib (b)		3	2	3		1
Class normal Ic (c)			5	1		2
Class Branches (d)		3	1	2		2
Class Rings (e)				1	8	
Class Spots (f)				1		4

Table 3.1: Misclassification matrix for the six-classes problem of a human expert

	classified as (a)	classified as (b)	classified as (c)	classified as (d)	classified as (e)	classified as (f)
Class normal Ia (a)	7	4	1			
Class normal Ib (b)	5	3	1			
Class normal Ic (c)		1	3	4		
Class Branches (d)		2	5	1		
Class Rings (e)	1				7	1
Class Spots (f)	1				2	2

Table 3.2: Misclassification matrix for the six-classes problem of C4.5

4.3 Results of other machine learning algorithms

We tried also the other machine learning algorithms on all the data sets. For testing we used the standard 10-fold cross validation, as described above. The results are given in Tables 4 and 5. Parametrization algorithms PCA and Association rules need a small subset of images for defining the attributes (this subset, called pre-training subset, is subtracted from the training set of images). We run each of these two algorithms on ten different pre-training subsets of images, so that we obtained for each of them ten different parametrizations. All machine learning algorithms were run on all of them and in tables we give the best results (the highest accuracy among ten average accuracies obtained from cross validations over all ten parametrizations). Note also that SMO can be used only for two-classes problems, therefore it is omitted from Table 5.

	C4.5	Naïve Bayes	K-NN	Kernel Density	SMO	Neural networks
IP	80.2 %	82.7 %	78.7%	76.2 %	84.7 %	83.1 %
PCA*	72.8 %	74.2 %	71.2 %	66.0 %	74.3 %	74.3 %
Assoc.rules *	80.9 %	75.1 %	80.0 %	76.8 %	83.8 %	83.9 %
GDV Assist	81.4 %	81.2 %	74.5 %%	65.7 %	80.6 %	76.9 %
GDV Assist with add. atts	81.7 %	80.9 %	72.6 %	64.6 %	80.5 %	71.9 %

Table 4: Classification accuracy of five machine learning algorithms on three different descriptions of coronas for two-classes problem (* we give best results over ten different pre-training subsets of images for Association rules and PCA parametrization algorithms)

	C4.5	Naïve Bayes	K-NN	Kernel Density	Neural networks
IP	60.2 %	55.4 %	56.8 %	55.5 %	65.2 %
PCA*	47.7 %	50.7 %	50.2 %	53.8 %	52.1 %
Assoc.rules *	50.9 %	37.5 %	51.5 %	52.5 %	61.6 %
GDV Assist	55.0 %	54.2 %	51.6 %	49.4 %	59.0 %
GDV Assist with add. atts	54.1 %	51.2 %	48.8 %	48.5 %	54.6 %

Table 5: Classification accuracy of five machine learning algorithms on three different descriptions of coronas for six-classes problem (* we give best results over ten different pre-training subsets of images for Association rules and PCA parametrization algorithms)

On the two-classes problem SMO (based on SVM machine learning algorithm) achieved the best results: accuracy was up to 85%. Neural networks were rather close with accuracy up to 84% using Associative rules for parametrization. The worse was Kernel Density algorithm, while the others achieved comparable results.

GDV Assistant, Image Processor with statistical parametrization, and parametrization based on

Association rules provide comparably good description of images, while the Principal Component Analysis provides the worst parametrization. This is somehow disappointing as we expected PCA to be more appropriate for describing coronas than algorithms, which are designed specially for textures. However, it is well known that PCA requires normalized images (coronas should have been all of approximately equal size and centered, pictures should all be of equal level of brightness) which was not the case in our study.

On the six-classes problem neural networks seems to perform best and achieved accuracy up to 65% using statistical parametrization of images. The other algorithms (without SMO, which cannot deal with more than two classes) achieved lower accuracy. Image Processor with statistical parametrization shows clear advantage over the other parametrization methods, which give comparable quality of image descriptions.

5. Conclusions and future work

We described the development of computer classifiers for various types of coronas. In this study we were interested in the implementation of computer programs for detection of coronas of people in altered states of consciousness. Such coronas are known to have rings (double coronas), special branch-like structure of streamers, and/or curious spots. Besides detecting altered states of consciousness (two-classes problem) we were interested also in classification of different types of coronas (six-classes problem).

We used several approaches to parametrization of images: statistical approach, principal component analysis, association rules and GDV software approach extended with several additional parameter. For the development of the classifiers we used various machine learning algorithms: learning of decision trees, naïve Bayesian classifier, K-nearest neighbors classifier, Support vector machine, and Kernel Density classifier.

The most competitive machine learning algorithms for our problem seem to be neural networks and Support vector machines (SVM incorporated in SMO algorithm). The worst seems to be the Kernel Density classifier.

Among parametrization techniques GDV Assistant, Image Processor (statistical approach) and Associative rules (symbolic approach) achieved similar quality of image descriptions, while the Principal Component Analysis was the worse. GDV Assistant seems to be a promising approach, however further tests and further improvements are necessary.

The best results (classification accuracy up to 85%) in the two-classes problem were achieved by SMO algorithm, based on Support vector machines, and using statistical parametrization of images. This result is better than that of a human expert, who achieved 77% of classification accuracy. In the six-classes problem, the best results (classification accuracy up to 65%) were achieved by neural networks with statistical parametrization of images. This result also outperforms that of the human expert. Those results indicate that computer algorithms can achieve the same or even better accuracy than human experts.

In future we plan to:
- add to GDV Assistant some more possibly informative parameters for this problem,
- we plan to try some other machine learning algorithms (other variants of decision trees and Naïve Bayes),
- test more human experts to see how accurate the manual classification can be,
- combine various parametrization techniques in order to extract possibly different and useful information from different sets of parameters and preprocess is using various feature subset selection approaches,
- combine the decisions of different classifiers in order to improve their reliability and try also other approaches to improving the classification accuracy, such as bagging, boosting, stacking, and transduction,

- we plan to collect additional images of coronas in order to increase the number of training/testing instances and therefore to improve the reliability of classifiers.

Acknowledgments
We thank dr. Konstantin Korotkov for providing the coronas of fingertips of people in altered states of consciousness. We thank also Vida Korenc for preprocessing the images and Ivan Vidmar for implementing additional parameters in GDV Assistant and for performing preliminary experiments with C4.5.

References

Aha, D., and D. Kibler (1991) "Instance-based learning algorithms", Machine Learning, vol.6, pp. 37-66.

R. Agrawal, T. Imielinski, and A. Swami (1993) Mining association rules between sets of items in large databases. In P. Buneman and S. Jajodia, editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Mangement of Data*, pages 207-216, Washington, D.C., 1993.

R. Agrawal and R. Srikant (1994) Fast algorithms for mining association rules. In J. B. Bocca, M. Jarke, and C. Zaniolo, editors, Proc. 20th Int. Conf. Very Large Data Bases, VLDB, pages 487-499. Morgan Kaufmann.

Bevk M. (2003) Texture Analysis with Machine Learning, M.Sc. Thesis, University of Ljubljana, Faculty of Computer and Information Science, Ljubljana, Slovenia. (in Slovene)

Julesz, B., Gilbert, E.N., Shepp, L.A., Frisch H.L.(1973). Inability of Humans To Discriminate Between Visual Textures That Agree in Second-Order-Statistics, Perception 2, pp. 391-405.

M. Bevk and I. Kononenko (2002) A statistical approach to texture description: A preliminary study. In ICML-2002 Workshop on Machine Learning in Computer Vision, pages 39-48, Sydney, Australia, 2002.

R. Haralick, K. Shanmugam, and I. Dinstein (1973) *Textural features for image classification*. IEEE Transactions on Systems, Man and Cybernetics, pages 610-621.

G. H. John and P. Langley (1995). Estimating Continuous Distributions in Bayesian Classifiers. Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence. pp. 338-345. Morgan Kaufmann, San Mateo.

S.S. Keerthi, S.K. Shevade, C. Bhattacharyya, K.R.K. Murthy (2001). Improvements to Platt's SMO Algorithm for SVM Classifier Design. Neural Computation, 13(3), pp 637-649, 2001.

Korotkov, K. (1998) *Aura and Consciousness*, St.Petersburg, Russia: State Editing & Publishing Unit “Kultura”.

Korotkov, K., Korotkin, D. (2001) Concentration dependence of gas discharge around drops of inorganic electrolytes, *Journal of Applied Physics*, Vol. 89, pp. 4732-4736.

J. Platt (1998). Fast Training of Support Vector Machines using Sequential Minimal Optimization. Advances in Kernel Methods - Support Vector Learning, B. Schölkopf, C. Burges, and A. Smola, eds., MIT Press.

W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling (1992) *Numerical Recipes: The Art of Scientific Computing*. Cambridge University Press, Cambridge (UK) and New York, 2nd edition.

J.R. Quinlan (1993) C4.5 Programs for Machine Learning, Morgan Kaufmann.

D.E. Rumelhart, G.E. Hinton, R.J. Williams (1986) Learning internal representations by error propagation. In: Rumelhart D.E. and McClelland J.L. (eds.) Parallel Distributed Processing, Vol. 1: Foundations. Cambridge: MIT Press.

J. A. Rushing, H. S. Ranagath, T. H. Hinke, and S. J. Graves (2001) Using association rules as texture features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 845-858.

A. Sadikov (2002) Computer visualization, parametrization and analysis of images of electrical gas discgarge (in Slovene), M.Sc. Thesis, University of Ljubljana, 2002.

L. Sirovich and M. Kirby (1987) A low-dimensional procedure for the characterisation of human faces. Journal of the Optical Society of America, pages 519-524.

A. Sadikov, I. Kononenko, F. Weibel (2003) Analyzing Coronas of Fruits and Leaves, This volume.

M. Turk and A. Pentland (1991) Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, pages 71-86.

I. H. Witten, E. Frank (2000) Data mining: Practical machine learning tools and techniques with Java implementations, Morgan Kaufmann.