Automatic Sleep Staging using Support Vector Machines with Posterior Probability Estimates

Steinn Gudmundsson
Dept. of Computer Science
University of Iceland
Reykjavik, Iceland

Thomas Philip Runarsson
Science Institute
University of Iceland
Reykjavik, Iceland
E-mail: tpr@hi.is

Sven Sigurdsson
Dept. of Computer Science
University of Iceland
Reykjavik, Iceland

Abstract

This paper describes attempts at constructing an automatic sleep stage classifier using EEG recordings. Three different feature extraction schemes were compared together with two different pattern classifiers, the recently introduced support vector machine and the well known k-nearest neighbor classifier. Using estimates of posterior probabilities for each of the sleep stages it was possible to devise a simple post-processing rule which leads to improved accuracy. Compared to a human expert the accuracy of the best classifier is 81%.

1 Introduction

Polysomnography (PSG) is a recording of various physiologic parameters during sleep and is used for diagnosis of sleep related disorders. The parameters of interest include e.g. the electroencephalogram (EEG), electrooculogram (EOG), electromyogram (EMG), respiratory activity. During analysis an overnight recording is divided into 20-30 second epochs and a sleep stage is assigned to each epoch according to the rules of Rechtschaffen and Kales [10] (R&K) which require at least one EEG channel, two EOG channels and an EMG channel. There are six different sleep stages, wake (W), light sleep (stages I and II), deep sleep (stages III and IV) and rapid eye movement (REM.) The time evolution of sleep in terms of sleep stages is called a *hypnogram* and is used for diagnosis. Sleep scoring is performed by an expert and is both difficult and time consuming. Automating the process would therefore be of great interest. Construction of automatic procedures has turned out to be difficult because the R&K rules are somewhat subjective (e.g. scoring of stages I and III) resulting in low inter scorer agreement and more importantly due to the fact that sleep does not appear to be a discrete process [9]. Despite of these shortcomings R&K continues to be the "gold standard" in sleep clinics.

The support vector machine (SVM) has recently become a popular approach to classification of data. SVMs have turned out to be useful in a wide variety of real-world classification problems delivering state-of-the-art performance. The aim of this study was to investigate how SVMs can be used to classify sleep stages on the basis of EEG alone. Furthermore, it will be shown how posterior probability estimates can be used to enhance classification accuracy.

The reminder of the paper is organized as follows. The next section describes the data set used in the experiments. Section 3 describes the feature extraction. A short overview of classification and support vector machines is given in section 4. Results are presented in section 5 and section 6 concludes the paper.

2 Data set

The data set consists of four over-night recordings of young subjects (mean age 5 years) which were recorded at the Helsinki University Hospital, Finland. The recordings have been scored by an expert neurologist according to the R&K rules using an epoch length of 30 seconds. EEG data from a single epoch together with it's label (sleep stage) constitutes one example in the data set.

Since only few examples are available from stage III, they are combined with examples from stage IV and labelled slow wave sleep (SWS). Furthermore, examples from stages I and II are combined and labelled light sleep (LS). Table 1 shows the distribution of class labels. It is obvious that the data set is not balanced, LS epochs make up roughly half of the examples. The EEGs were recorded using the Nervus system (Taugagreining hf, Iceland) at a sampling rate of 256 Hz. A single EEG channel, C3-A2 was used. The data was band pass filtered in the range 0.5 - 70 Hz and a 50 Hz notch filter applied to remove mains interference.

Subject	W	LS	SWS	REM
S52	124	549	198	157
S62	136	567	218	136
S71	123	487	299	167
S73	86	554	192	129
Total (4122)	469	2157	907	589

Table 1. Number of examples per class.

3 Feature extraction

Instead of using the raw time series as input to a pattern classifier, several features (attributes) are extracted from the EEG of each epoch. These features are then used as inputs to the classifier. The problem of feature selection, i.e. which features to use is non-trivial, ideally the features should be relatively few and describe the time series accurately. Many different features have been proposed in the EEG literature such as Hjorth complexity parameters [6], features derived from the power spectrum [3], autoregressive modelling [9] and theory of nonlinear dynamical systems [3]. Here the Hjorth complexity parameters and several features based on the power spectrum are described along with a new feature based on histograms.

3.1 Hjorth complexity parameters

Hjorth [6] provides three quantitative descriptors of EEG called activity, mobility and complexity. For each epoch these measures are computed as follows: Activity $= \sigma_0$, Mobility $= \sigma_1/\sigma_0$, Complexity $= \sigma_2/\sigma_1$ where σ_i denotes the variance of the i-th derivative of the signal (the 0-th derivative corresponds to the signal itself.) All three parameters form the input to the classifier.

3.2 Features from power spectrum

An estimate of the power spectrum is computed for each epoch using Welch's averaged periodogram method [8]. Relative power in the following frequency bands was computed, 0.5 - 2 Hz, 2 - 4 Hz, 4 - 5 Hz, 5 - 7 Hz, 7 - 10 Hz, 10 - 13 Hz, 13 - 15 Hz, 15 - 20 Hz, 20 - 30 Hz and 30 - 40 Hz. In addition the median frequency (MF) of the signal and spectral entropy (SEN) were computed $SEN = -(1/\log N)\sum_{k=1}^{N} s_k \log s_k$ where s_k is the normalized power spectrum in frequency bin k and N is the number of bins. The spectral entropy is a measure of the regularity of the signal, a pure sine wave has entropy zero and uncorrelated white noise has entropy one. The ten relative power values together with MF and SEN form the input to the classifier.

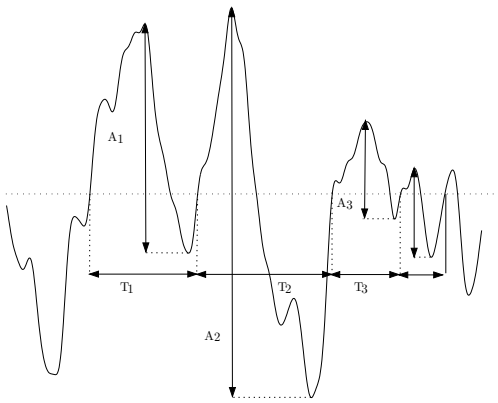

Figure 1. Analysis of zero-crossings.

3.3 Histogram features based on waveform measures

A two dimensional histogram of the amplitude and frequency distribution of a single EEG epoch is constructed and used as input to the classifier. The frequency-amplitude histogram is constructed by analyzing successive zero crossings of the signal as illustrated in figure 1. The time between successive zero crossings is denoted by T_i and the peak-to-peak amplitude A_i. The bin count is incremented for each pair (T_i, A_i). The partitioning of the histogram is loosely based on the R&K rules. The frequency axis is split into ten intervals corresponding to the frequency bands from the previous section. The amplitude axis is split into five intervals: below 5 μV, 5 - 30 μV, 30 - 75 μV, 75 - 100 μV and 100 - 400 μV.

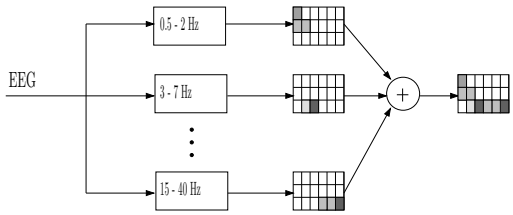

Figure 2. Bank of filters.

In order to capture fast-wave activity which may be superimposed on slower activity (e.g. sleep spindles), the EEG is sent through a bank of band-pass filters (3rd order Butterworth.) A histogram is constructed from the output of each filter and all the histograms combined into a single histogram which serves as input to the pattern classifier (see figure 2.) Definition of the filter pass bands is based on the traditional EEG frequency bands, δ (0.5 - 2 Hz), θ (3 - 7 Hz), α (8 - 12 Hz), σ (12 - 14 Hz) and $\beta+$ (15 - 40 Hz).

4 Classification

The pattern recognition problem can be stated as follows: Assume that there are M different classes of objects, given a new object assign it to one of the M classes. Each object is associated with certain measurements $\mathbf{x}$ which form the *feature vector*. The set of possible classes is denoted by $\mathcal{Y}$. In supervised learning a training data set $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_\ell)$ with $y_i \in \mathcal{Y}$ is given and the task is to construct a decision rule (classifier) that assigns to a new measurement $\mathbf{x}$ a label $y \in \mathcal{Y}$. In the following it is assumed that $\mathbf{x}_i \subset \mathbb{R}^d$ and $\mathcal{Y} = \{\text{Wake}, \text{LS}, \text{SWS}, \text{REM}\}$.

Once a classifier has been trained it is necessary to evaluate its performance on an independent (labelled) test set. This is done by presenting the test examples one at a time to the classifier and counting the errors. Assume that R misclassifications are made on a test set of size ℓ_T. The probability of misclassification, p is estimated by $\hat{p} = R/\ell_T$ (the estimated accuracy is $1 - \hat{p}$.) It is easy to show [11] that a 95% confidence interval on $\hat{p}$ is given by

$$\hat{p} \pm 2\sqrt{\hat{p}(1-\hat{p})/\ell_T} \tag{1}$$

When working with a single data set *cross-validation* is often used to evaluate classifier accuracy. The original data set is split into v disjoint subsets of equal size ℓ/v. The classifier is trained v times. Each time a different subset is left out for testing and the classifier trained on the remaining $v - 1$ subsets. The average accuracy over all the test sets is an estimate of the true classifier accuracy.

To get a more accurate estimate how the classifiers perform on unseen recordings a variation of the cross-validation scheme described earlier is used. One subject is removed from the data set at a time and used as a test set, the remaining subjects are merged into a single training set which is used to construct a classifier. Because the examples from each subject are somewhat correlated, using standard cross validation would not give an independent test set and result in overly optimistic estimates of the classifier accuracy.

4.1 Nearest neighbor classifiers

The k-NN classifier is known to perform well on many practical problems [7] and is used as a benchmark in this study. The classifier works as follows: For a given measurement $\mathbf{x}$ assign to it the label most frequently represented amongst the k nearest examples in the training set. An appropriately scaled Euclidian distance metric is used as a measure of distance between examples. Ties are broken arbitrarily.

4.2 Support vector machines

Recently so-called kernel methods have become popular in various data mining applications. One example is support vector machines (SVMs) which have proven useful in many practical classification problems [2]. To begin with assume that there are only two classes, $\mathcal{Y} = \{-1, +1\}$. The basic idea behind SVM classifiers is to introduce a mapping $\mathbf{x} \to \phi(\mathbf{x})$ that maps the data into a linear space $\mathcal{H}$ where they are (almost) linearly separable[1] and then use a linear classifier.[2]

For linearly separable data it can be shown that a maximum margin hyperplane is optimal with respect to generalization properties (i.e. ability to classify unseen data.) The maximum margin hyperplane separates the two classes so that the distance from the hyperplane to the closest example(s) is maximal. These examples are known as *support vectors*. One of the most important properties of SVMs is that the mapping ϕ need not to be explicitly known, only the inner product (kernel) $K(\mathbf{x}_1, \mathbf{x}_2) = \langle \phi(\mathbf{x}_1), \phi(\mathbf{x}_2) \rangle$ is needed. The choice of kernel is problem specific but the radial basis function (RBF) kernel is commonly used

$$K(\mathbf{x}_1, \mathbf{x}_2) = \exp\left(-\gamma ||\mathbf{x}_1 - \mathbf{x}_2||^2\right)$$

where γ is a free parameter to be specified. If the data is not linearly separable in $\mathcal{H}$ the optimization problem is modified so that incorrect classifications are allowed at a cost. Finding the maximum margin subject to such a modification can be formulated as a quadratic optimization problem (QP)

$$\begin{array}{ll} \min_{\mathbf{w},b,\boldsymbol{\xi}} & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{\ell}\xi_i \\ \text{subject to} & y_i(\mathbf{w}^T\phi(\mathbf{x}_i) + b) \geq 1 - \xi_i \\ & \xi_i \geq 0, \qquad i = 1, \ldots, \ell \end{array}$$

where C denotes the relative weight between the two competing objectives. The classification of a test example $\mathbf{x}$ is performed by computing $\text{sign}(f(\mathbf{x}))$, $f(\mathbf{x}) = \sum_{i=1}^{\ell} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b$ where α_i are the Lagrangian multipliers for the dual problem formulation. Solving the dual problem avoids the need for computing the features $\phi(\mathbf{x})$ explicitly.

Extension to multi-class problems can be done in several ways. The strategy used here follows [1]. Construct $M(M-1)/2$ binary classifiers each trained on data from two of the specified classes. A point $\mathbf{x}$ is sent through all the classifiers. Each binary classification is considered to be a voting, the class label that gets the highest number of votes is then assigned to the point $\mathbf{x}$. Ties are broken arbitrarily. This is the so-called one-against-one (pairwise) approach for multi-class classification.

[1] i.e. a hyperplane exists so that all the $+1$ examples lie on on side of the plane and all the -1 examples lie on the other side.

[2] The decision rule is a linear function of $\mathbf{x}$.

In order to solve the QP problem(s) the parameters, C and γ have to be specified. Proper selection of the parameters is essential for good performance and is discussed section 5. The SVM experiments were carried out using the LIBSVM package [1].

4.3 Posterior probabilities

Given an example $\mathbf{x}$ both k-NN and SVM simply assign $\mathbf{x}$ to one of the classes W,LS,DS and REM. In many applications it is desirable to get an estimate of the posterior probabilities for each class in addition to the classification. Given a measurement $\mathbf{x}$ the goal is to estimate

$$p_i = p(y = i \mid \mathbf{x}), \qquad i = 1, \ldots, M$$

In the case that all the posteriors are almost equal, it might be wise to "reject" the classification, on the other hand if a single posterior probability dominates all other, the confidence in the classification is higher.

For the k-NN classifier a well known estimate of the posterior probabilities is [4]

$$p_i = p(y = i \mid \mathbf{x}) \approx k_i/k, \qquad i = 1, \ldots, M$$

where k_i is the number of examples amongst the k nearest neighbors that belong to class i. Note that for small values of k some of the k_i may be zero.

The LIBSVM package estimates posterior probabilities by fitting a sigmoid function that maps SVM outputs f to posterior probabilities. First the pairwise class probabilities are estimated

$$r_{ij} = p(y = i \mid y = i \text{ or } j, \mathbf{x}) \approx \frac{1}{1 + \exp(Af + B)}$$

where A and B are estimated by minimizing the negative log-likelihood function using known training data and their decision values f. Five fold cross validation is used to obtain decision values because labels and decision values are required to be independent. Once all the r_{ij}'s have been obtained, the p_i's are obtained by solving the following optimization problem

$$\begin{array}{ll} \min_{\mathbf{p}} & \frac{1}{2}\sum_{i=1}^{M}\sum_{j\neq i}(r_{ji}p_i - r_{ij}p_j)^2 \\ \text{subject to} & \sum_{i=1}^{M} p_i = 1 \\ & p_i \geq 0, \qquad i = 1, \ldots, M \end{array}$$

See [1] and references therein for more details.

Once the posterior probability estimates are available they can be used to implement a simple post processing rule for the sleep stage classification. If the maximum posterior probability (over all stages) for a given epoch is below a pre-specified threshold, p_{min}, do not accept change in sleep stage (i.e. use the sleep stage from the previous epoch.) This procedure may result in improved classification due to "inertia" which is built into the R&K rules.

5 Results

In general it is necessary to scale the training data prior to training the classifier. For LIBSVM, linear scaling into the interval $[-1, 1]$ is recommended for the RBF kernel [1]. The test set is then scaled accordingly. The same scaling strategy is used for the k-NN classifier.

It still remains to select parameters (C, γ) for the SVM and the number of neighbors k for the k-NN. For the SVM the parameters are selected for each training set using a five-fold cross validation. An exhaustive (grid) search over a large set of (C, γ) pairs is performed and the pair with the highest cross- validation accuracy selected. The SVM classifier is then retrained using all the data in the training set with the optimal parameter values and the performance evaluated using the test set (i.e. a single subject.) To take an example of when the accuracy is evaluated for recording S52. A training set is constructed by merging data from recordings S62, S71 and S73. Five-fold cross validation on this training set is used to find optimal values of C and γ (SVM). Using the optimal parameters the SVM classifier is retrained and tested on data from recording S52. The same procedure is carried out for the 30-NN classifier except no parameters had to be optimized. The number of neighbors for k-NN could be varied and selecting the value giving the highest cross validation accuracy on each training set. Because the optimal value of k might turn out to be quite small the posterior probability estimates would be inaccurate. Since we are interested in the effect of the posterior probability estimates k is always set to 30 and cross validation accuracy estimated from the training set. The results are almost identical to using the optimal value of k. In the following the number of neighbors is therefore fixed, $k = 30$. For misclassification probabilities in the range 0.15 - 0.35 as we have in the tables below on a single test set ($\ell_T \approx 1000$), equation (1) shows that $\hat{p}$ is known to within 0.03 at a 95% confidence level.

5.1 Hjorth's complexity parameters

Prior to computing the Hjorth parameters a 30 Hz high-cut filter (6th order Butterworth) is applied to the data because the parameters are sensitive to noise. Table 2 shows the accuracy, number of support vectors (#SV) and optimal values of C and γ for the SVM together with results for the nearest neighbor algorithm. The SVM does slightly better than 30-NN.

5.2 Power spectral measures

Table 3 shows the accuracy obtained using the power spectrum features. The performance of the SVM is very similar to 30-NN. The accuracy is slightly higher than was

Test set	(C^*, γ^*)	SVM #SV	Accuracy	30-NN Accuracy
S52	$(2^{15}, 2^1)$	1320	0.76	0.69
S62	$(2^{15}, 2^1)$	1301	0.61	0.60
S71	$(2^{13}, 2^1)$	1257	0.69	0.70
S73	$(2^{11}, 2^3)$	1428	0.76	0.77
Overall			0.71	0.69

Table 2. Results using the Hjorth complexity measures.

obtained using the Hjorth feature set. The SVM has fewer support vectors than before, indicating that the power spectrum measures are better at discriminating between sleep stages.

Test set	(C^*, γ^*)	SVM #SV	Accuracy	30-NN Accuracy
S52	$(2^{13}, 2^{-3})$	991	0.73	0.75
S62	$(2^{7}, 2^{-1})$	1065	0.71	0.76
S71	$(2^{13}, 2^{-3})$	928	0.73	0.65
S73	$(2^{11}, 2^{-1})$	1057	0.84	0.83
Overall			0.75	0.75

Table 3. Results using the power spectrum features.

5.3 Waveform measures

Table 4 shows the accuracy obtained using the histogram features based on waveform measures. The accuracy is similar to what was obtained using the power spectrum features. the 30-NN classifier performs slightly better than SVM. The main reason is the failure of the latter on S62. The values of (C, γ) found using cross validation on S52, S71 and S73 do not work well for classification of S62. One explanation is that the complete data set is relatively small, i.e. more than three recordings are needed to find optimal values of (C, γ).

Test set	(C^*, γ^*)	SVM #SV	Accuracy	k-NN Accuracy
S52	$(2^{11}, 2^{-7})$	886	0.76	0.77
S62	$(2^{3}, 2^{-3})$	940	0.67	0.76
S71	$(2^{1}, 2^{-1})$	1001	0.78	0.78
S73	$(2^{1}, 2^{-1})$	1079	0.82	0.82
Overall			0.76	0.79

Table 4. Results using the histogram method.

5.4 Post-processing

Table 5 shows the accuracy obtained using the standard SVM classifier (std) and the one with a threshold (thr) on the posterior probabilities, $p_{\min} = 0.7$ (arbitrarily chosen.) The corresponding values for the k-NN classifier are given in table 6. In all cases but one the post processing improves the accuracy slightly. Figure 3 shows an actual hypnogram (top) together with automatically generated hypnograms using the standard SVM classifier (middle) and SVM classifier with threshold (bottom). The post processing appears to capture the "inertia" built into the R&K rules to some extent, i.e. it reduces the rapid switching between stages seen in the plain SVM classifier.

Additional insight into the performance of a classifier is obtained by constructing a *confusion matrix* which shows the relation between actual class counts and predicted class counts. The matrix has one row and one column for each class. An element in row i and column j counts the number of times class i was classified as j. Diagonal elements count the number of correct classifications and off-diagonal elements count the number of misclassifications. Two confusion matrices for the SVM classifier with threshold are shown in tables 7 - 8. There is considerable confusion of REM stages with LS indicating the need to add EOG and/or EMG information. There is also some confusion of W and LS stages which is not surprising because the latter includes sleep stage I which can be hard to distinguish from the wake state [9]. Finally there is some confusion between LS and DS which might be attributed to the fact that the latter includes sleep stage III which is known to overlap sleep stage II [9].

Test set	Hjorth std	Hjorth thr	PSD std	PSD thr	Histo. std	Histo. thr
S52	0.76	0.72	0.73	0.73	0.76	0.77
S62	0.61	0.63	0.71	0.71	0.67	0.78
S71	0.69	0.69	0.73	0.74	0.78	0.81
S73	0.76	0.78	0.84	0.88	0.82	0.86
Overall	0.71	0.71	0.75	0.77	0.76	0.81

Table 5. Standard versus threshold for SVM.

Test set	Hjorth std	Hjorth thr	PSD std	PSD thr	Histo. std	Histo. thr
S52	0.68	0.72	0.75	0.77	0.76	0.80
S62	0.60	0.60	0.76	0.76	0.77	0.77
S71	0.70	0.70	0.65	0.68	0.79	0.77
S73	0.77	0.79	0.83	0.84	0.83	0.85
Overall	0.69	0.70	0.75	0.76	0.79	0.80

Table 6. Standard versus threshold for 30-NN.

	W	LS	SWS	REM
W	92	29	1	1
LS	5	468	7	7
SWS	0	48	251	0
REM	1	126	0	40

Table 7. Confusion matrix for S71.

	W	LS	SWS	REM
W	71	10	2	3
LS	59	438	21	36
SWS	16	5	171	0
REM	2	5	0	122

Table 8. Confusion matrix for S73.

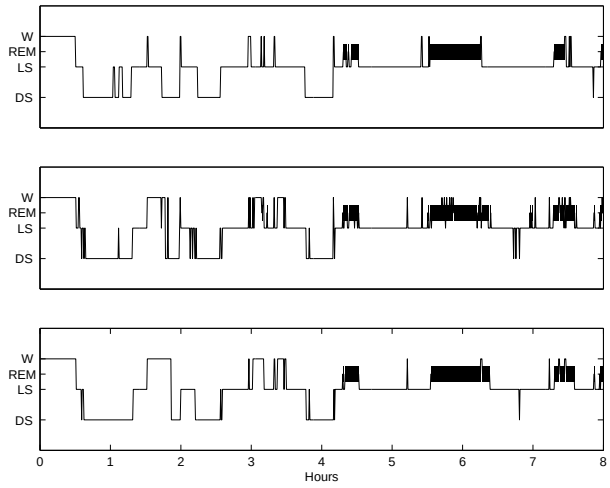

Figure 3. Hypnograms for S73. Expert scoring (top), output of SVM (middle) and SVM with threshold (bottom).

6 Summary

Of the the three feature sets tested, the histogram based on waveform measures gave the best results, the power spectrum features came in second and the Hjorth parameters third. The simple k-NN classifier gave results comparable to the SVM suggesting that feature extraction is the critical issue. Using a threshold for the posterior probability estimates enhances the accuracy. In this case the best accuracy was 81% which might be high enough to be clinically useful. Studies on (visual) inter-scorer agreement using the full set of R&K rules report 67% - 91% agreement, see [5] and references therein for more details. Experts might thus use hypnograms generated by an automatic classifier as a starting point and manually refine as needed, saving considerable time. Future work aims at including EMG and EOG data in order to get improved detection of REM stages and decrease confusion between LS and wake. Furthermore, there is room for improvement when it comes to extraction of EEG features.

Acknowledgments

We are grateful to Kimmo Sainio, MD, PhD at the Department of Children, University Hospital, Helsinki, Finland for providing the sleep recordings. The project was supported by The Icelandic Center for Research (RANNIS).

References

[1] Chang, C.C. and Lin, C.J. LIBSVM: a library for support vector machines, 2001 Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm

[2] Cristianini, N. and Shawe-Taylor, J. (2000) *An introduction to support vector machines*, Cambridge University Press

[3] Fell, J., Röschke, J., Mann K. and Schäffner C. Discrimination of sleep stages: a comparison between spectral and nonlinear EEG measures, Electroenceph. clin. Neurophysiol., vol. 98, pp. 401-410, 1996

[4] Fukunaga, K. and Hostetler, L. k-nearest-neighbor Bayes-risk estimation, IEEE Trans. Information Theory, Vol. 21, No. 3, pp. 285-293, May 1975

[5] Agarwal, R. and Gotman, J. Computer-assisted sleep staging, IEEE Transactions on Biomedical Engineering, 48, No. 12, December 2001

[6] Hjorth, B. (1975) Time domain descriptors and their relation to a particular model for generation of EEG activity, in *CEAN - Computerized EEG analysis*, p. 3-8, Gustav Fischer Verlag

[7] Michie, D., Spiegelhalter, D. J. and Taylor, C. C. (eds) (1994) *Machine Learning, Neural and Statistical Classification*, Ellis Horwood

[8] Oppenheim, A. V. and Schafer, R.W. (1999) *Discrete-time signal processing*, Prentice Hall

[9] Pardey, J., Roberts, S., Tarassenko, L. and Stradling, J. A new approach to the analysis of the human sleep/wakefulness continuum, J. Sleep Res., **5**(4), 201-210, 1996

[10] Rechtschaffen, A. and Kales, A. (1968) *A manual of standardized terminology techniques and scoring system for sleep stages of human subjects*. Brain Reasearch Institute, UCLA, Los Angeles, USA

[11] Ripley, B.D. (1996) *Pattern recognition and neural networks*, Cambridge University Press